On Applications of New Soft and Evolutionary Computing Techniques to Direct and Inverse Modeling Problems

Thesis submitted in partial fulfillment of the requirements for the award of the Doctor of Philosophy

by

Babita Majhi

Roll No. 50609003, Ph. D.

Under the guidance of

Prof. (Dr.) G. Panda, FNAE, FNASc.

Electronics & Communication Engineering

National Institute of Technology

Rourkela - 769008

CERTIFICATE

This is to certify that the thesis entitled "**On Applications of New Soft and Evolutionary Computing Techniques to Direct and Inverse Modeling Problems**" by Ms. Babita Majhi, submitted to the National Institute of Technology, Rourkela for the degree of Doctor of Philosophy, is a record of bonafide research work carried out by her in the department of Electronics and Communication Engineering under my supervision. I believe that the thesis fulfills part of the requirements for the award of degree of Doctor of Philosophy. The results embodied in the thesis have not been submitted for award of any other degree.

Dr. Ganapati Panda, FNAE, FNASc.

Professor
Department of ECE
National Institute of Technology
Rourkela – 769008
India

ACKNOWLEDGEMENT

I am indebted to many people who contributed through their support, knowledge and friendship, to this work and the years at NIT Rourkela.

I am grateful to my supervisor, Prof. G. Panda, who gave me the opportunity to realize this work in the laboratory. He encouraged, supported and motivated me with much kindness throughout the work. I always had the freedom to follow my own ideas, which I am very grateful for. I really admire him for patience and staying power to carefully read the whole thesis. It is his help for which I stand where I am.

I am also grateful to NIT Rourkela for providing me adequate infrastructure to carry out the present investigations.

I am thankful to Prof. K. K. Mahapatra, Prof. S. K. Patra, Prof. S. Meher of Electronics and Communication Engg. department and Prof. U. K. Mohanty of Metallurgical and Materials Engg. department for extending their valuable suggestions and help whenever I approached.

My special thanks to Dr. D. P. Acharya and Ajit Kumar Sahoo for their constant inspiration and encouragement during my research.

My hearty thanks to Sitanshu, Jagganath, Trilochan, Upendra, Sudhansu, Pyari, Nithin, Vikas, Pawan, Sasmita and Piter for their help, cooperation and encouragement.

I acknowledge all staff, research scholars and juniors of ECE department, NIT Rourkela for helping me.

I render my respect to all my family members for giving me mental support and inspiration for carrying out my research work.

(Babita Majhi)
Roll. No. 50609003, Ph. D.

ABSTRACT

Adaptive direct modeling or system identification and adaptive inverse modeling or channel equalization find extensive applications in telecommunication, control system, instrumentation, power system engineering and geophysics. If the plants or systems are nonlinear, dynamic , Hammerstein and multiple-input and multiple-output (MIMO) types, the identification task becomes very difficult.

Further, the existing conventional methods like the least mean square (LMS) and recursive least square (RLS) algorithms do not provide satisfactory training to develop accurate direct and inverse models. Very often these (LMS and RLS) derivative based algorithms do not lead to optimal solutions in pole-zero and Hammerstein type system identification problem as they have tendency to be trapped by local minima.

In many practical situations the output data are contaminated with impulsive type outliers in addition to measurement noise. The density of the outliers may be up to 50%, which means that about 50% of the available data are affected by outliers. The strength of these outliers may be two to five times the maximum amplitude of the signal. Under such adverse conditions the available learning algorithms are not effective in imparting satisfactory training to update the weights of the adaptive models. As a result the resultant direct and inverse models become inaccurate and improper.

Hence there are three important issues which need attention to be resolved. These are :

(i) Development of accurate direct and inverse models of complex plants using some novel architecture and new learning techniques.

(ii) Development of new training rules which alleviates local minima problem during training and thus help in generating improved adaptive models.

(iii) Development of robust training strategy which is less sensitive to outliers in training and thus to create identification and equalization models which are robust against outliers.

These issues are addressed in this thesis and corresponding contribution are outlined in seven Chapters. In addition, one Chapter on introduction, another on required architectures and

algorithms and last Chapter on conclusion and scope for further research work are embodied in the thesis.

A new cascaded low complexity functional link artificial neural network (FLANN) structure is proposed and the corresponding learning algorithm is derived and used to identify nonlinear dynamic plants. In terms of identification performance this model is shown to outperform the multilayer perceptron and FLANN model. A novel method of identification of IIR plants is proposed using comprehensive learning particle swarm optimization (CLPSO) algorithm. It is shown that the new approach is more accurate in identification and takes less CPU time compared to those obtained by existing recursive LMS (RLMS), genetic algorithm (GA) and PSO based approaches. The bacterial foraging optimization (BFO) and PSO are used to develop efficient learning algorithms to train models to identify nonlinear dynamic and MIMO plants. The new scheme takes less computational effort, more accurate and consumes less input samples for training. Robust identification and equalization of complex plants have been carried out using outliers in training sets through minimization of robust norms using PSO and BFO based methods. This method yields robust performance both in equalization and identification tasks. Identification of Hammerstein plants has been achieved successfully using PSO, new clonal PSO (CPSO) and immunized PSO (IPSO) algorithms. Finally the thesis proposes a distributed approach to identification of plants by developing two distributed learning algorithms : incremental PSO and diffusion PSO. It is shown that the new approach is more efficient in terms of accuracy and training time compared to centralized PSO based approach. In addition a robust distributed approach for identification is proposed and its performance has been evaluated.

In essence the thesis proposed many new and efficient algorithms and structure for identification and equalization task such as distributed algorithms, robust algorithms, algorithms for ploe-zero identification and Hammerstein models. All these new methods are shown to be better in terms of performance, speed of computation or accuracy of results.

Contents

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ADSL	Adaptive digital subscriber loop
AIS	Artificial immune system
AWGN	Additive white Gaussian noise
BER	Bit error ratio
BFO	Bacterial foraging optimization
BIBO	Bounded input bounded output
BP	Back propagation
CF	Cost function
CFLANN	Cascaded functional link artificial neural network
CLPSO	Comprehensive learning particle swarm optimization
CNN	Chebyshev neural network
CPSO	Clonal particle swarm optimization
DPSO	Diffusion particle swarm optimization
DSP	Digital signal processing
EVR	Eigen value ratio
FE	Functional expansion
FIR	Finite impulse response
FLANN	Functional link artificial neural network
FPGA	Field programmable gate array
GA	Genetic Algorithm
HVAC	Heating, ventilating and air conditioning
IIR	Infinite impulse response
ILMS	IIR LMS
INPSO	Incremental particle swarm optimization
IPSO	Immunized particle swarm optimization
ISI	Inter Symbol Interference
LMS	Least mean square
MGS	Mackey glass system
MIMO	Multiple input multiple output
MLANN	Multilayer artificial neural network
MLP	Multilayer perceptron
MLSE	Mean log squared error
MMSE	Minimum mean square error
MSE	Mean square error
NMSE	Normalized mean square error
PPN	Polynomial perceptron network
PSO	Particle swarm optimization
RBF	Radial basis function
RCF	Robust cost function
RLMS	Recursive least mean square
RLS	Recursive least square
RNN	Recurrent neural network
SI	Swarm intelligence
SISO	Single input single output
SSE	Sum squared error

SSE	Sum squared errors
VLSI	Very large scale integrated
WNN	Wavelet neural network

Chapter

1

Introduction

1.1 Background

OUT of many applications of adaptive filtering, direct modeling and inverse modeling are very important. The direct modeling or system identification finds applications in control system engineering including robotics [1.1], intelligent sensor design [1.2], process control [1.3], power system engineering [1.4], image and speech processing [1.4], geophysics [1.5], acoustic noise and vibration control [1.6] and biomedical engineering [1.7]. Similarly inverse modeling technique is used in digital data reconstruction [1.8], channel equalization in digital communication [1.9], digital magnetic data recording [1.10], intelligent sensor [1.2], deconvolution of seismic data [1.11]. The direct modeling mainly refers to adaptive identification of unknown plants. Simple static linear plants are easily identified through parameter estimation using conventional derivative based least mean square (LMS) type algorithms [1.12]. But most of the practical plants are dynamic, nonlinear and combination

of these two characteristics. In many applications Hammerstein and MIMO plants need identification. In addition the output of the plant is associated with measurement or additive white Gaussian noise(AWGN). Identification of such complex plants is a difficult task and poses many challenging problems. Similarly inverse modeling of telecommunication and magnetic medium channels is also important for reducing the effect of inter symbol interference (ISI) and achieving faithful reconstruction of original data. Similarly adaptive inverse modeling of sensors is required to extend their linearities for direct digital readout and enhancement of dynamic range. If the channel or the sensor characteristic is modeled as a nonlinear filter with large eigen-value ratio (EVR) together with AWGN building up of an accurate inverse model is also a difficult and challenging task. These two important and complex issues are addressed in the thesis and attempts have been made to provide improved efficient and alternate promising solutions.

The conventional LMS and recursive least square (RLS) [1.13] techniques work well for identification of static plants but when the plants are of dynamic type, the existing forward-backward LMS [1.14] and the RLS algorithms very often lead to non optimal solution due to premature convergence of weights to local minima [1.15]. This is a major drawback of the use of existing derivative based techniques. To alleviate this burning issue this thesis suggests the use of derivative free optimization techniques in place of conventional techniques.

In recent past population based optimization techniques have been reported which fall under the category of evolutionary computing [1.16] or computational intelligence [1.17]. These are also called bio-inspired techniques which include genetic algorithm (GA) and its variants [1.18], particle swarm optimization (PSO) and its variants [1.19], bacterial foraging optimization (BFO) and its variants [1.20] and artificial immune system (AIS) and its variants [1.21]. These techniques are suitably employed to obtain efficient iterative learning algorithms for developing adaptive direct and inverse models of complex plants and channels.

Development of direct and inverse adaptive models essentially consists of two components. The first component is an adaptive network which may be linear

or nonlinear in nature. Use of a nonlinear network is preferable when nonlinear plants or channels are to be identified or equalized. The linear networks used in the thesis are adaptive linear combiner or all-zero or FIR structure [1.7] and pole-zero or IIR structure[1.7]. Under nonlinear category low complexity single layer function link artificial neural network (FLANN) [1.22] and multilayer perceptron network (MLP) [1.23] are used. The second component is the training or learning algorithm used to train the parameters of the model. As stated earlier the structures used are trained by bio-inspired techniques such as GA, PSO and modified PSOs, BFO and modified BFOs. Depending upon the complexity and nature of the plants to be identified proper combination of network of the model and corresponding bio-inspired learning rule is selected so that the combination yields the best possible performance in direct and inverse modeling tasks. This requires the knowledge of prior experience and simulation results. One of the objectives of the present investigation is to choose models with appropriate combination of structure and algorithm so that it provides best possible performance of direct and inverse models. The bio-inspired optimization tools can not directly be applied to develop direct and inverse models of plants as those are not aimed to be used for training of parameters of models. Therefore another motivation of investigation is to formulate the direct and inverse modeling problems as optimization problems and then to introduce bio-inspired techniques suitably to effectively optimize the cost function of the models. In conventional identification and equalization problems, the mean square error at the output is considered as the cost function to be minimized by using bio-inspired techniques.

In many practical situations the training signal available is highly corrupted by outliers and may be as high as 50%. Under such constraints the training of the models gets severely affected if the squared error is used as the cost function for minimization. This is because this conventional cost function is not robust against outliers [1.24]. In statistics few cost functions have been defined which are robust in nature and are not affected by outliers. These are Wilcoxon norm, $\sigma(1-\exp(-e^2/2\sigma))$ and $\log(1+\frac{e^2}{2})$, where σ is a parameter to be adjusted during

training and e^2 is mean square error. In this thesis robust identification, equalization and time series prediction schemes by minimization of the robust norm using bio-inspired techniques have been proposed. This is a novel contribution in this thesis.

In recent years distributed signal processing has played an important role in sensor networks in which individual node collects local information but the objective is to compute the global solution. Some representative problems are parameter estimation using locally measured data and nonlinear identification using the local data in a cooperative manner. Attempts have been made to solve this interesting problem by using an approach based on newly introduced distributed PSO. In this work two distributed versions: incremental and diffusion type PSO techniques have been proposed and then used for robust identification of linear and nonlinear plants.

1.2 Motivation

In summary the main motivations of the research work carried in the present thesis are the following :

(i) To formulate the direct and inverse modeling problems as error square optimization problems

(ii) To introduce bio-inspired optimization tools such as PSO and BFO and their variants to efficiently minimize the squared error cost function of the models. In other words to develop alternate identification scheme.

(iii) To achieve improved identification (direct modeling) of complex nonlinear all-zero, pole-zero, Hammerstein and MIMO plants and channel equalization (inverse modeling) of nonlinear noisy digital channels by introducing new and improved identification algorithms.

(iv) To devise new bio-inspired training strategy for robust identification of complex plants and robust equalization of complex channels.

(v) To suggest distributed incremental and diffusion type PSO algorithms and use them for identification of linear and nonlinear plants using local data of each sensor node.

(vi) To introduce distributed robust algorithms for identification of nonlinear plants.

1.3 Major contribution of the thesis

The following novel contributions have been made in the thesis :

A low complexity functional link artificial neural network based nonlinear dynamic system identifier has been developed and its learning algorithm has been derived. Improved identification performance has been demonstrated through simulation study.

A comprehensive learning particle swarm optimization technique has been used to effectively identify IIR plants. Further, extensive simulation study has been made on the use of the proposed method to effectively identify higher order plants with lower order models. The new approach has been shown to overcome local minima problem in a multimodal situation.

The bacterial foraging optimization and particle swarm optimization have been used as learning tools in developing new models for identification of dynamic systems. Robust identification and prediction task has also been carried using PSO. Similarly a new approach to develop robust inverse model in presence of outliers has been successfully implemented using BFO.

Identification of complex Hammerstein plants using two new PSO algorithms : clonal PSO and immunized PSO have been proposed. It is shown that the immunized PSO model outperforms its counterpart in all counts.

Distributed incremental and diffusion type PSO algorithms have been suggested for identification of nonlinear plants with outliers in the training signal under a sensor network frame work. The results are observed to be superior to conventional method of identification.

1.4 Chapter wise contribution

The research work undertaken is embodied in 10 Chapters.

1. Introduction
2. Selected adaptive architectures and bio-inspired techniques, principles and algorithms
3. Development of a new cascaded functional link artificial neural network(CFLANN) for nonlinear dynamic system identification
4. Identification of IIR plants using comprehensive learning particle swarm optimization
5. Dynamic systems identification using PSO and BFO based learning algorithms.
6. Robust identification and prediction using particle swarm optimization technique
7. Robust adaptive inverse modeling using bacterial foraging optimization technique and applications
8. Identification of Hammerstein plants using Clonal PSO and Immunized PSO algorithms
9. Development of distributed PSO algorithms for robust nonlinear system identification
10. Conclusion and scope for further work

Out of 10 Chapters, the research contribution is contained in Chapters 3 to 9. A brief outline of chapter wise contribution is presented in sequel. **Chapter 1** outlines the introduction to the problem, the motivation of the research work and a condensed version of chapter wise contribution made in the thesis. Finally **Chapter 10** deals with the overall conclusion of the investigation and scope for further research work.

A brief outline of each of the linear and nonlinear networks used for identification and equalization purpose is presented in **Chapter 2.** This

includes all-zero and pole-zero adaptive filters under linear category and MLP, FLANN, CNN under nonlinear category. This Chapter also reviews the existing derivative based algorithms such as the LMS, recursive LMS (RLMS), FLANN, BP and evolutionary computing optimization algorithms like the GA, PSO, BFO and AIS. Various combinations of the structure and the learning algorithms are suitably used to obtain novel adaptive models for identification and equalization of complex plants and channels respectively.

In **Chapter 3**, a new cascaded FLANN structure is proposed and the corresponding learning algorithm is derived and used to identify nonlinear dynamic plants. Four different identification models have been suggested. Results of identification through simulation demonstrate that the new models outperform the conventional MLP and FLANN based models in terms of computational load and accuracy.

Identification of IIR plants or pole-zero systems finds extensive applications in echo cancellation, channel estimation, process control, array processing and speech recognition. In **Chapter 4** a new adaptive IIR algorithm using comprehensive learning particle swarm optimization (CLPSO) is proposed which avoids potential local minima problem and provides accurate estimate of pole-zero coefficients of IIR plants. Simulation study of identification of some benchmark IIR plants reveals that the proposed method outperforms the existing recursive LMS (RLMS), GA and PSO based methods in terms of mean square error(MSE), execution time and product of population size and number of input samples used during training.

Chapter 5 deals with the development of PSO and BFO based schemes to identify nonlinear dynamic single-input-single-output (SISO) and multiple-input-multiple-output (MIMO) systems. The BFO and PSO based training of the weights of the FLANN identification model have been newly introduced. In both cases it is observed that the new training schemes in complex identification task work better in terms of speed of computation, accuracy and number of input samples used for training. However, both the new schemes offer almost similar identification performance.

The problem of robust identification and prediction is studied in depth in **Chapter 6**. Development of such identification scheme is required when either the training or input signal samples are contaminated with outliers. The existing squared error cost function based learning schemes fail to offer satisfactory identification performance. Therefore the use of new robust norms which are insensitive to outliers have been suggested as the cost function. Such cost functions are minimized by PSO method to develop robust identification of nonlinear dynamic systems and prediction models of complex time series. Simulation results show that the Wilcoxon norm produces the best robust models for identification and prediction compared to that produced by other norms used in the investigation.

Inverse modeling plays an important role in channel equalization, sensor linearization and deconvolution operation in geophysics applications. In practice it is difficult to develop an inverse model using squared error norm when the training signal contains outliers. Therefore investigation has been made in **Chapter 7** to identify robust norms of errors and use them to develop robust inverse models. To obtain such models the robust norms are minimized using BFO scheme. The robustness of the new inverse models is evaluated through simulation study using some benchmark channels and different percentage of outliers in the training signal. The results indicate that the use of squared error provides the least robust inverse models where as the Wilcoxon norm generates most robust models.

Hammerstein plant contains all features of complex plants because it contains a static nonlinear part, a dynamic linear system and an additive coloured noise. Identification of such complex plants really poses difficulty. In **Chapter 8**, attempt has been made to identify such plants using two new PSO algorithms : clonal PSO (CPSO) and immunized PSO (IPSO). In this Chapter two new variants of PSO algorithm have been proposed and then used them in training the model parameters. The potentiality of the proposed method is demonstrated through simulation of benchmark Hammerstein plants. The potentiality of identification is evaluated by verifying three features : the response matching at the output of static nonlinear part, comparison of

estimated parameters of linear dynamic part with the corresponding true values and comparison of sum of squared errors (SSE) between true and overall estimated responses. Comparing all these test results, it is observed that the IPSO model outperforms its counterpart in all counts.

In application like sensor networks linear and nonlinear identifications are required using local data of each sensor. To achieve this objective distributed signal processing algorithm for training is required. In **Chapter 9** two such distributed algorithms known as incremental and diffusion PSO (INPSO and DPSO) algorithms have been proposed to identify linear and nonlinear plants. Further the squared and Wilcoxon norm of errors are minimized in a cooperative manner using distributed PSO algorithms. Simulation results demonstrate that both the distributed algorithms provide excellent identification performance when conventional squared error norm is used. When outliers are present in the training samples both the Wilcoxon norm based distributed algorithms provide superior performance compared to the conventional norm based training.

The overall conclusion of the total investigation is listed in **Chapter 10**. This Chapter also contains the details of further research work that can be carried out in the same or the related field.

References

[1.1] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", IEEE Trans. on Neural Networks, vol. 1, pp. 4-26, January 1990.

[1.2] J. C. Patra, A. C. Kot and G. Panda, "An intelligent pressure sensor using neural networks", IEEE Trans. on Instrumentation and Measurement, vol. 49, issue 4, pp. 829-834, Aug. 2000.

[1.3] M. Pachter and O. R. Reynolds, "Identification of a discrete time dynamical system", IEEE Trans. Aerospace Electronic System, vol. 36, issue 1, pp. 212-225, 2000.

[1.4] G. B. Giannakis and E. Serpedin, "A bibliography on nonlinear system identification", Signal Processing, vol. 83, no. 3, pp. 533-580, 2001.

[1.5] E. A. Robinson and T. Durrani, *Geophysical Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1986.

[1.6] D. P. Das and G. Panda, "Active mitigation of nonlinear noise processes using a novel filtered-s LMS algorithm", IEEE Trans. on Speech and Audio Processing, vol. 12, issue 3, pp. 313-322, May 2004.

[1.7] B. Widrow and S.D. Sterns, "*Adaptive Signal Processing"* Prentice-Hall, Inc. Engle-wood Cliffs, New Jersey, 1985.

[1.8] G. J. Gibson, S. Siu and C. F. N. Cowan, "The application of nonlinear structures to the reconstruction of binary signals", IEEE Trans. signal processing, vol. 39, no. 8, pp. 1877-1884, Aug. 1991.

[1.9] R. W. Lucky, Techniques for adaptive equalization of digital communication systems, Bell Sys.Tech. J., 45, 255-286, Feb. 1966.

[1.10] H. Sun, G. Mathew and B. Farhang-Boroujeny, "Detection techniques for high density magnetic recording", IEEE Trans. on Magnetics, vol. 41, no. 3, pp. 1193-1199, March 2005.

[1.11] L. J. Griffiths, F. R. Smolka and L. D. Trenbly, "Adaptive deconvolution : a new technique for processing time varying seismic data", Geohysics, June 1977.

[1.12] B. Widrow, J. M. McCool, M. G. Larimore and C. R. Johnson, Jr., :Stationary and nonstationary learning characteristics of the LMS adaptive filter", Proc. IEEE, vol. 64, no. 8, pp. 1151-1162, Aug., 1976.

[1.13] B. Friedlander and M. Morf, "Least-squares algorithms for adaptive linear phase filtering", IEEE Trans., vol. ASSP-30, no. 3, pp. 381-390, June 1982.

[1.14] S. A. White, "An adaptive recursive digital filter", Proc. 9th Asilomar Conf. Circuits Syst. Comput., p. 21, Nov. 1975.

[1.15] John J. Shynk, "Adaptive IIR filtering", IEEE ASSP Magazine, April 1989, pp. 4-21.

[1.16] A. E. Eiben and J. E. Smith, "*Introduction to Evolutionary Computing*", Springer, 2003, ISBN 3-540-40184-9.

[1.17] Andries Engelbrecht, "*Computational Intelligence : An introduction"*, Wiley & Sons, ISBN 0-470-84870-7.

[1.18] D.E.Goldberg, "*Genetic algorithms in search, optimization and machine learning*", Addition-Wesley,1989.

[1.19] J. Kennedy, R. C. Eberhart and Y. Shi, "*Swarm intelligence",* San Francisco: Morgan Kaufmann Publishers, 2001.

[1.20] K. M. Passino, "Biomimicry of Bacterial Foraging for distributed optimization and control", IEEE control system magazine, vol 22, issue 3, pp. 52-67, June 2002.

[1.21] D. Dasgupta, *Artificial Immune Systems and their Applications*, Springer-Verlag, 1999.

[1.22] Y.H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison Wesley, Reading, Massachusetts, 1989.

[1.23] S. Haykin, "*Neural Networks: A comprehensive foundation*" 2nd Edition, Pearson Education Asia, 2002.

[1.24] Jer-Guang Hsieh, Yih-Lon Lin and Jyh-Horng Jeng, "Preliminary study on Wilcoxon learning machines", IEEE Trans. on neural networks, vol. 19, no. 2, pp. 201-211, Feb. 2008.

Chapter

2

Selected Adaptive Architectures and Bio-Inspired Techniques, Principles and Algorithms

2.1 Introduction

THE main motive of the research work carried out in this thesis is to develop elegant and efficient adaptive identification schemes for complex nonlinear and dynamic plants, adaptive inverse models of nonlinear plants, equalization of complex channels and prediction of nonlinear time series. All these adaptive models inherently need suitable adaptive structures and appropriate learning rules to train the parameters of these models. In the present investigation, I briefly outline some selected adaptive architectures such as adaptive linear combiner, adaptive pole-zero filters, functional link artificial neural network and multilayer artificial neural network. In addition we present some recently developed population based bio-inspired derivative free techniques such as particle swarm optimization and its

variants, bacterial foraging optimization and its variants and artificial immune system and its variants for training the parameters or coefficients of the adaptive structures. An adaptive linear combiner or filter is feed forward in structure. These are [2.1, 2.2] often realized either as a set of program instructions running on an arithmetical processing device such as a microprocessor or DSP chip, or as a set of logic operations implemented in a field-programmable gate array (FPGA) or in a semi-custom or custom Very large scale integrated (VLSI) circuit. An adaptive linear combiner is characterized by

1. the input signal sampled employed,
2. the structure that defines how the output signal of the combiner is computed from its input samples,
3. the parameters of the structure which are iteratively changed based on some learning rule and
4. the adaptive algorithms that guide how the parameters are to be adjusted iteratively until the predefined objective is fulfilled.

By choosing a particular adaptive filter structure, one specifies the number and type of parameters that need adjustments. The adaptive algorithms used to update these parameters tend to minimize the cost function of the model. The cost function normally are mean squared error of the model which is not robust to outliers in the training or desired signal samples. The thesis also introduces minimization of some robust cost functions of the error for updating the model parameters. Essentially the development of adaptive models for identification, equalization, prediction and function approximation is viewed as a minimization problem of some suitable cost functions. The main contribution of the thesis is to solve these complex optimization problems using bio-inspired based learning rules.

In following section, the general adaptive filtering problem is presented and the mathematical notation for representing the form and operation of the adaptive filter is introduced.

2.2 The adaptive filtering problem

Figure 2.1 shows a block diagram of an adaptive FIR filter or an adaptive linear combiner in which a sample from a digital input signal x_k is fed into an adaptive filter, that computes a corresponding output signal sample y_k at time k. The output signal is compared to a second signal d_k, called the desired signal.

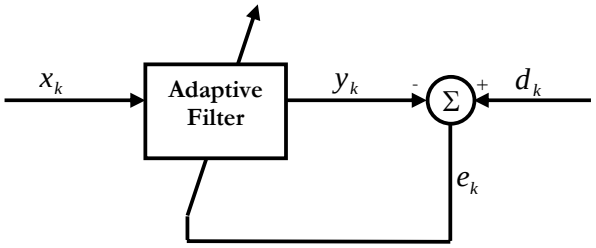


Fig. 2.1 The general adaptive filtering problem

The difference signal given by

$$e_k = d_k - y_k \tag{2.1}$$

is known as the error signal. The error signal is used to adapt the parameters of the filter from time k to time $(k+1)$ in a well-defined manner. This process of adaptation is represented by an oblique arrow. As the time index k is incremented the output of the adaptive filter matches a better and better to the desired signal following an adaptation process such that the magnitude of e_k decreases over time.

In the adaptive filtering framework, adaptation refers to the mechanism by which the parameters of the system are changed from time index k to time index $(k+1)$. The number and types of parameters within this system depend on the computational structure chosen for the system. Different filter structures that have been used for model development are presented below.

2. 2. 1 Adaptive FIR filter

The general architecture of an FIR adaptive filter or a adaptive linear combiner [2.1] is depicted in Fig. 2.2. Let X is N^{th} input pattern having one unit delay in each instant. This process is also called as adaptive linear combiner [2.1-2.2]. Let $X_n = [x(n) \quad x(n-1)..... x(n-M+1)]$ form of the M -by-1 tap input vector and $M-1$ is the number of delay elements. The tap weights $W_n = [w_0(n) \quad w_1(n) ..... w_{M-1}(n)]^T$ form the elements of the M -by-1 tap weight vector. The output is represented as,

$$y(n) = \sum_{m=0}^{M-1} w_m(n) x(n-m) \tag{2.2}$$

The output can be represented in vector notation as

$$y(n) = X_n^T W_n = W_n^T X_n \tag{2.3}$$

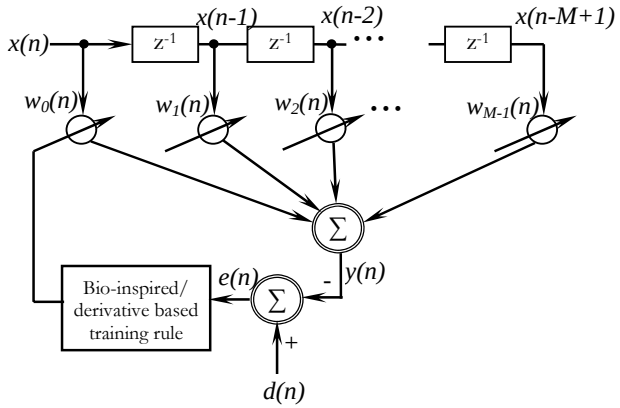


Fig. 2.2 Adaptive filter using Bio-inspired/Derivative based algorithms

2. 2. 2 Adaptive IIR filter

The structure of a direct-form adaptive IIR filter [2.1] is shown in Fig. 2.3. In this case, the output of the system is given by

$$y(n) = \sum_{m=1}^{N-1} a_m(n) y(n-m) + \sum_{m=0}^{M-1} b_m(n) x(n-m) \tag{2.4}$$

The terms $a_m(n)$ and $b_m(n)$ represent the feed forward and feed back coefficients of the filter respectively. In matrix form, $y(n)$ may be written as

$$y(n) = W_n^T S_n \tag{2.5}$$

Where the combined weight vector is

$$W_n = [b_0(n)\ b_1(n) \ldots\ldots\ldots\ldots b_{M-1}(n)\ \ a_1(n)\ \ a_2(n) \ldots\ldots\ldots\ldots a_{N-1}(n)]^T \tag{2.6}$$

And the combined input and output signal vector is

$$S_n = [x(n)\ \ x(n-1) \ldots\ldots\ldots x(n-M+1)\ \ y(n-1)\ \ y(n-2) \ldots\ldots\ldots y(n-N+1)]^T \tag{2.7}$$

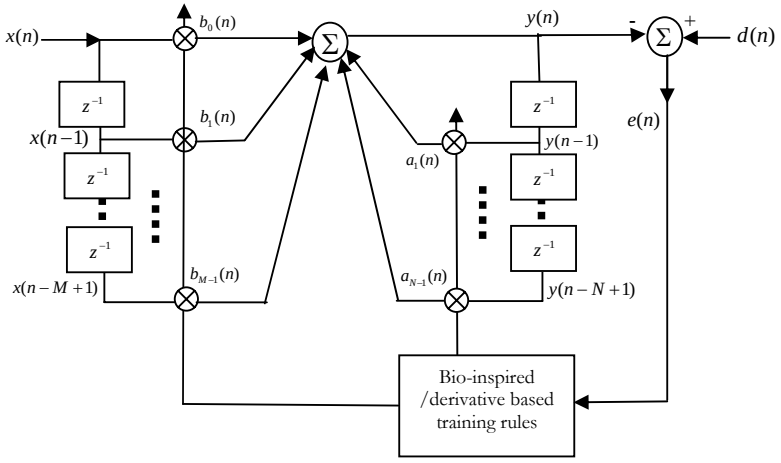


Fig. 2.3 Structure of an adaptive IIR filter

The weight update operation of adaptive IIR filter is carried out using either conventional derivative based or derivative free learning algorithms. In addition to the linear structures nonlinear structure can be used for which the principle of superposition does not hold when the parameter values are fixed. Such systems are useful when the relationship between $d(n)$ and $x(n)$ is not linear in nature. This class of nonlinear structure consists of artificial neural network (ANN), functional link artificial neural network (FLANN) and radial basis function (RBF) network. These networks inherently contain distributed nonlinear elements in each path like the sigmoid function in ANN, sine / cosine terms in FLANN and Gaussian function in RBF network. In the next section details of these nonlinear structures are dealt.

2.3 Artificial neural network (ANN)

Artificial neural network (ANN) takes its name from the network of nerve cells in the brain. Recently, ANN has proved to be an important technique for classification and optimization problems [2.3-2.5]. McCulloch and Pitts have developed the neural networks for different computing machines. There are extensive applications of various types of ANN in the field of communication, control, instrumentation and forecasting. The ANN is capable of performing nonlinear mapping between the input and output space due to its large parallel interconnection between different layers and the nonlinear processing characteristics. An artificial neuron basically consists of a computing element that performs the weighted sum of the input signal and the connecting weight. The sum is added with the bias or threshold and the resultant signal is then passed through a nonlinear function of sigmoid or hyperbolic tangent type. Each neuron is associated with three parameters whose learning can be adjusted; these are the connecting weights, the bias and the slope of the nonlinear function. For the structural point of view, a neural network (NN) may be single layer or it may be multilayer. In multilayer structure, there is one or many artificial neurons in each layer and for a practical case there may be a number of layers. Each neuron of the one layer is connected to each neuron of the next layer. The functional-link ANN is another type of single layer NN. In this type of network the input data is allowed to pass through a functional expansion block where the input data are nonlinearly mapped to more number of points. This is achieved by using trigonometric functions, tensor products or power terms of the input. The output of the functional expansion is then passed through a single neuron.

Twp types of NNs used in this thesis are discussed next.

2. 3. 1 Single neuron structure

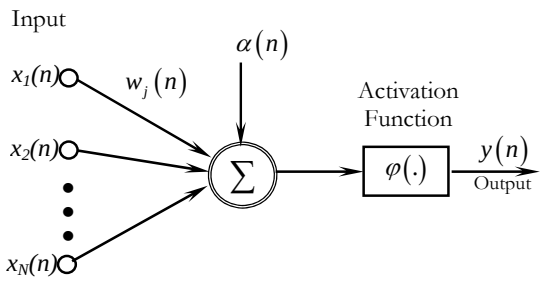


Fig. 2.4 Structure of a single neuron

The basic structure of an artificial neuron is presented in Fig. 2.4. The operation in a neuron involves the computation of the weighted sum of inputs and threshold [2.3-2.5]. The resultant signal is then passed through a nonlinear activation function. This is also called as a perceptron, which is built around a nonlinear neuron. The output of the neuron may be represented as,

$$y(n)=\varphi\left[\sum_{j=1}^{N} w_j(n)\,x_j(n)+\alpha(n)\right] \tag{2.8}$$

where $\alpha(n)$ is the threshold to the neurons at the first layer, $w_j(n)$ is the weight associated with the j^{th} input, N is the no. of inputs to the neuron and $\varphi(.)$ is the nonlinear activation function. Different types of nonlinear function are shown in Fig. 2.5.

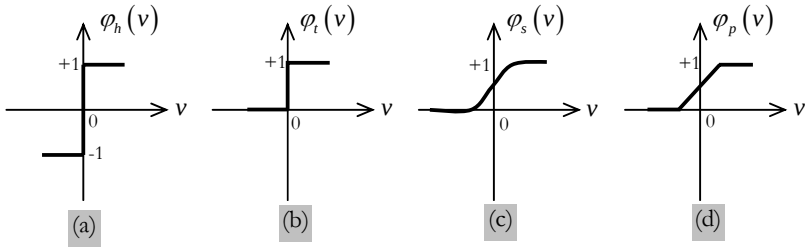


Fig. 2.5 Different types of nonlinear activation function,
(a) Signum function or hard limiter,
(b) Threshold function,
(c) Sigmoid function,
(d) Piecewise Linear

Signum Function: For this type of activation function, we have

$$\varphi(v)=\begin{cases}1 & if\ v>0\\ 0 & if\ v=0\\ -1 & if\ v<0\end{cases} \tag{2.9}$$

Threshold Function: This function is represented as,

$$\varphi(v)=\begin{cases}1 & if\ v\geq 0\\ 0 & if\ v<0\end{cases} \tag{2.10}$$

Sigmoid Function: This function is S-shaped and is the most common form of the activation function used in artificial neural network. It is a function that exhibits a graceful balance between linear and nonlinear behaviour.

$$\varphi(v) = \frac{1}{1+e^{-av}} \tag{2.11}$$

where v is the input to the sigmoid function and a is the slope of the sigmoid function. For the steady convergence a proper choice of a is required.

Piecewise-Linear Function: This function is

$$\varphi(v) = \begin{cases} 1, & v \geq +\frac{1}{2} \\ v, & +\frac{1}{2} > v > -\frac{1}{2} \\ 0, & v \leq +\frac{1}{2} \end{cases} \tag{2.12}$$

where the amplification factor inside the linear region of operation is assumed to be unity. Out of these nonlinear functions the sigmoid activation function is extensively used in ANN.

2. 3. 2 Multilayer perceptron (MLP)

In the multilayer neural network or multilayer perceptron (MLP), the input signal propagates through the network in a forward direction, on a layer-by-layer basis. This network has been applied successfully to solve some difficult and diverse problems by training in a supervised manner with a highly popular algorithm known as the error back-propagation algorithm [2.3, 2.4]. The scheme of MLP using four layers is shown in Fig. 2.6. $x_i(n)$ represent the input to the network, f_j and f_k represent the output of the two hidden layers and $y_l(n)$ represents the output of the final layer of the neural network. The connecting weights between the input to the first hidden layer, first to second hidden layer and the second hidden layer to the output layers are represented by w_{ij}, w_{jk} and w_{kl} respectively.

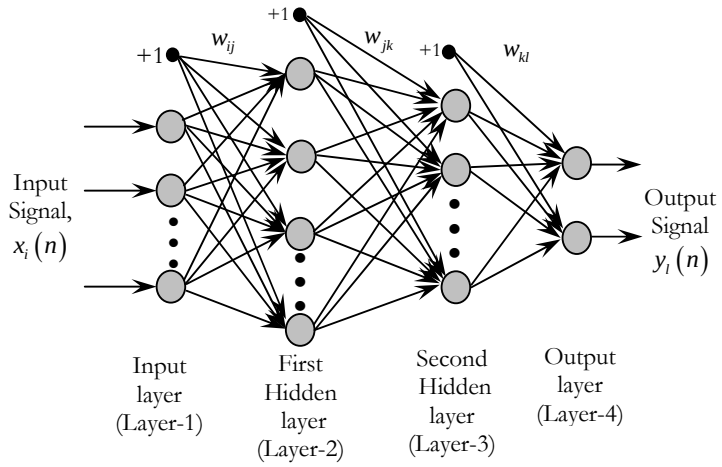


Fig. 2.6 MLP Structure

If P_1 is the number of neurons in the first hidden layer, each element of the output vector of first hidden layer may be calculated as,

$$f_j = \varphi_j\left[\sum_{i=1}^{N} w_{ij}x_i(n) + \alpha_j\right],\ \ i = 1,2,3,...N,\ j = 1,2,3,...P_1 \tag{2.13}$$

where α_j is the threshold to the neurons of the first hidden layer, N is the number of inputs and $\varphi(.)$ is the nonlinear activation function of the neurons of the first hidden layer which is defined in (2.11). The time index n has been dropped to make the equations simpler. Let P_2 be the number of neurons in the second hidden layer. The output of this layer is represented as, f_k and may be written as

$$f_k = \varphi_k\left[\sum_{j=1}^{P_1} w_{jk} f_j + \alpha_k\right],\quad k=1, 2, 3, \ldots, P_2 \tag{2.14}$$

where, α_k is the threshold to the neurons of the second hidden layer. The output of the final output layer can be calculated as

$$y_l(n) = \varphi_l\left[\sum_{k=1}^{P_2} w_{kl} f_k + \alpha_l\right], \quad l=1, 2, 3, \ldots, P_3 \tag{2.15}$$

where, α_l is the threshold to the neuron of the final layer and P_3 is the number of neurons in the output layer. The output of the MLP may be expressed as

$$y_l(n) = \varphi_n\left[\sum_{k=1}^{P_2} w_{kl}\varphi_k\left(\sum_{j=1}^{P_1} w_{jk}\varphi_j\left\{\sum_{i=1}^{N} w_{ij}x_i(n) + \alpha_j\right\} + \alpha_k\right) + \alpha_l\right] \tag{2.16}$$

The details of BP algorithm used to train the weights of various layers of the ANN are discussed in 2. 4. 1. 3.

2. 3. 3 Functional link artificial neural network (FLANN)

Pao originally proposed FLANN which is a novel single layer ANN structure capable of forming arbitrarily complex decision regions by generating nonlinear decision boundaries [2.6, 2.7]. In this structure, the initial representation of a pattern is enhanced by using nonlinear function and thus the pattern dimension space is increased. The functional link acts on an element of a pattern or entire pattern itself by generating a set of linearly independent function and then evaluates these functions with the pattern as the argument. Hence separation of the patterns becomes possible in the enhanced space. The use of FLANN not only increases the learning rate but also has less computational complexity [2.9]. Pao *et al* [2.8] have investigated the learning and generalization characteristics of a random vector FLANN and compared with those attainable with MLP structure trained with back propagation algorithm by taking few functional approximation problems. A FLANN structure with two inputs is shown in Fig. 2.7.

Let **X** is the input vector of size $N\times1$ which represents N number of elements; the n^{th} element is given by:

$$\mathbf{X}(n) = x_n, 1 \le n \le N \tag{2.17}$$

Each element undergoes nonlinear expansion to form M elements such that the resultant matrix has the dimension of $N\times M$.

The functional expansion of the element x_n by power series expansion is carried out using the equation given in (2.18)

$$s_i = \begin{cases} x_n & \text{for } i = 1 \\ x_n^l & \text{for } i = 2,3,4,\ldots,M \end{cases} \tag{2.18}$$

where $l = 1,2,\cdots,M$.

For trigonometric expansion, the expanded elements are

$$s_i = \begin{cases} x_n & \text{for } i = 1 \\ \sin\left(l\pi x_n\right) & \text{for } i = 2,4,\ldots,M \\ \cos\left(l\pi x_n\right) & \text{for } i = 3,5,\ldots,M+1 \end{cases} \tag{2.19}$$

where $l = 1,2,\cdots,M/2$.

For Chebyshev expansion the terms are given by

$T_0(x_n) = 1$ for $n = 0$

$T_1(x_n) = x_n$ for $n = 1$

$T_2(x_n) = 2x_n^2 - 1$ for $n = 2$

$$T_{n+1}(x_n) = 2x_n T_n(x_n) - T_{n-1}(x_n) \text{ for } n > 2 \tag{2.20}$$

In matrix notation the expanded elements of the input vector is denoted by **S** of size $N\times(M+1)$.

The bias input is unity. So an extra unity value is padded with the **S** matrix and the dimension of the **S** matrix becomes $N\times Q$, where $Q = \left(M + 2\right)$.

Let the weight vector is represented as **W** having Q elements. The output y is given as

$$y = \sum_{i=1}^{Q} s_i w_i \tag{2.21}$$

In matrix notation the output is written as

$$\mathbf{Y} = \mathbf{S} \cdot \mathbf{W}^{\mathrm{T}} \tag{2.22}$$

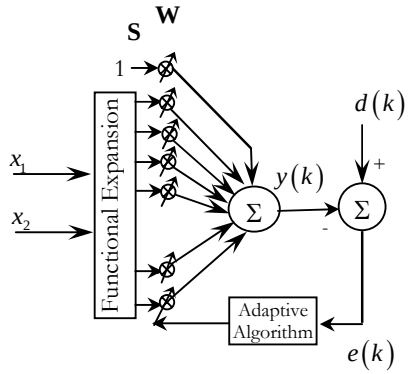


Fig. 2.7 Structure of the FLANN model

2. 4 Learning algorithms

There are many learning algorithms which are employed to train various adaptive models. The performance of these models depends on rate of convergence, training time, computational complexity involved and minimum mean square error achieved after training. The learning algorithms may be broadly classified into two categories (a) derivative based (b) derivative free. The derivative based algorithms include least means square(LMS), IIR LMS (ILMS), back propagation(BP) and FLANN-LMS. Under the derivative free algorithms, genetic algorithm(GA), particle swarm optimization(PSO), bacterial foraging optimization(BFO) and artificial immune system(AIS) have been employed. In this section the details of these algorithms are outlined in sequel.

2.4.1 Derivative based algorithms

These algorithms are gradient search in nature and have been derived by taking derivative of the squared error as the cost function. During the process of training these algorithms tend to drive the weights of the model to local minima. This leads to premature termination of the weights. As

a result the mean square error does not attain the least possible value and hence the accuracy of prediction becomes inferior. However these learning algorithms are simple to implement and can be expressed in close form equations. A brief description of each of them is presented below.

2.4.1.1 LMS algorithm for adaptive FIR filters

In an adaptive FIR filter, at any k th instant the error signal, e_k is computed as

$$e_k = d_k - y_k \tag{2.23}$$

where

d_k = the desired or training signal at k th instant

y_k = the output of the filter at k th instant

The weights associated with the filter are then updated using the LMS algorithm [2.1]. The weight updation equation for n^{th} instant is given by

$$w_k(n+1) = w_k(n) + \Delta w_k(n) \tag{2.24}$$

where $\Delta w_k(n)$ is the change of k th weight at n th iteration.

The change in weight of each path in each iteration is obtained by minimizing the mean squared error [2.1]. Using this value the weight update equation is given as

$$w_k(n+1) = w_k(n) + 2 \cdot \eta \cdot e_k(n) \cdot \mathbf{X}_k^{\mathrm{T}} \tag{2.25}$$

where η is the learning rate parameter ($0 \leq \eta \leq 1$). This procedure is repeated till the mean square error (MSE) of the network approaches a minimum value. The MSE at the time index k may be defined as, $\xi = E\left[e_k^2\right]$, where $E[.]$ is the expectation value or average of the signal.

2.4.1.2 Adaptive IIR LMS (ILMS) algorithm

Referring to Fig. 2.3, the error term is given by

$$e_k = d_k - y_k \tag{2.26}$$

where y_k is obtained from (2.4) and (2.5). The ILMS update rule is given as

$$W_{k+1} = W_k - M\ \hat{\nabla}_k \tag{2.27}$$

where $\hat{\nabla}_k$ = estimate of the gradient at k th instant and is given by

$$\hat{\nabla}_k = -2e_k[\alpha_{0k}.....\alpha_{Nk} \quad \beta_{1k}.....\beta_{Nk}]^T \tag{2.28}$$

M is a diagonal matrix of $N+1$ learning parameters for zero coefficients and N parameters for pole coefficients and is represented as

$$M = diag[\mu.....\mu \quad \eta.....\eta] \tag{2.29}$$

The variables α_{nk} and β_{nk} are given by

$$\alpha_{nk} = \frac{\partial y_k}{\partial a_n} = x_{k-n} + \sum_{l=1}^{L} b_l\ \alpha_{n,k-l}; \qquad 0 \le n \le L \tag{2.30}$$

$$\beta_{nk} = \frac{\partial y_k}{\partial b_n} = y_{k-n} + \sum_{l=1}^{L} b_l\ \beta_{n,k-l}; \qquad 1 \le n \le L \tag{2.31}$$

Equations (2.26) to (2.31) represent the key equations of ILMS algorithm.

2.4.1.3 Back propagation (BP) algorithm

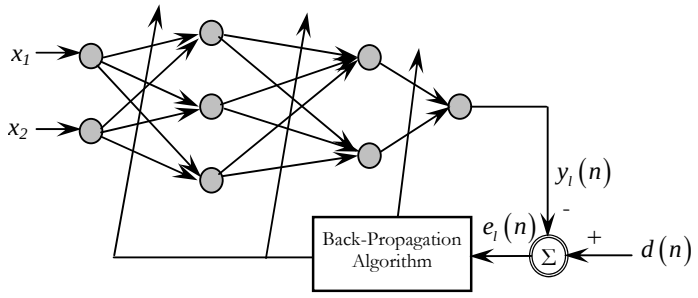


Fig. 2.8 Neural network using BP algorithm

The generalized structure of MLP is shown in Fig. 2.6. To derive the BP algorithm a simplified neural network with two inputs and 2-3-2-1 neurons (2, 3, 2 and 1 denote the number of neurons in the input layer, the first hidden layer, the second hidden layer and the output layer respectively) is depicted in Fig. 2.8. The parameters of the neural network can be updated in both sequential and batch mode of operation. In BP algorithm, the weights and the thresholds are initialized as very small random values. The intermediate and the final outputs of the MLP are calculated by using (2.13), (2.14), and (2.15) respectively.

The final output $y_l(n)$ at the output of neuron l, is compared with the desired output $d(n)$ and the resulting error signal $e_l(n)$ is obtained as

$$e_l(n) = d(n) - y_l(n) \tag{2.32}$$

The instantaneous value of the total error energy is obtained by summing all errors squared over all neurons in the output layer, that is

$$\xi(n) = \frac{1}{2}\sum_{l=1}^{P_3} e_l^2(n) \tag{2.33}$$

where P_3 is the no. of neurons in the output layer.

This error signal is used to update the weights and thresholds of the hidden layers as well as the output layer. The reflected error components at each of the hidden layers is computed using the errors of the last layer and the connecting weights between the hidden and the last layer and error obtained at this stage is used to update the weights between the input and the hidden layer. The thresholds are also updated in a similar manner as that of the corresponding connecting weights. The weights and the thresholds are updated in an iterative method until the error signal becomes minimum.

The weights are updated according to the following equations

$$w_{kl}(n+1) = w_{kl}(n) + \Delta w_{kl}(n) \tag{2.34}$$

$$w_{jk}(n+1) = w_{jk}(n) + \Delta w_{jk}(n) \tag{2.35}$$

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n) \tag{2.36}$$

where, $\Delta w_{kl}(n), \Delta w_{jk}(n)$ and $\Delta w_{ij}(n)$ are the change in weights of the second hidden layer-to-output layer, first hidden layer-to-second hidden layer and input layer-to-first hidden layer respectively. This can be computed as

$$\begin{aligned} \Delta w_{kl}(n) &= -2\mu \frac{d\xi(n)}{dw_{kl}(n)} = 2\mu e(n) \frac{dy_l(n)}{dw_{kl}(n)} \\ &= 2\mu e(n) \varphi_l' \left[\sum_{k=1}^{P_2} w_{kl} f_k + \alpha_l \right] f_k \end{aligned} \tag{2.37}$$

where, μ is the convergence coefficient ($0 \le \mu \le 1$). Similarly $\Delta w_{jk}(n)$ and $\Delta w_{ij}(n)$ can also be computed as

The thresholds of each layer can be updated in a similar manner using equations

$$\alpha_l(n+1) = \alpha_l(n) + \Delta\alpha_l(n) \tag{2.38}$$

$$\alpha_k(n+1) = \alpha_k(n) + \Delta\alpha_k(n) \tag{2.39}$$

$$\alpha_j(n+1) = \alpha_j(n) + \Delta\alpha_j(n) \tag{2.40}$$

where, $\Delta\alpha_l(n), \Delta\alpha_k(n)$ and $\Delta\alpha_j(n)$ are the change in thresholds of the output, hidden and input layer respectively. The change in threshold is represented as,

$$\begin{aligned} \Delta\alpha_l(n) &= -2\mu \frac{d\xi(n)}{d\alpha_l(n)} = 2\mu e(n) \frac{dy_l(n)}{d\alpha_l(n)} \\ &= 2\mu e(n) \varphi_l' \left[\sum_{k=1}^{P_2} w_{kl} f_k + \alpha_l \right] \end{aligned} \tag{2.41}$$

2.4.1.4 The FLANN algorithm

Referring the structure of the FLANN of Fig. 2.7, the error signal $e(k)$ at k^{th} iteration can be computed as

$$e(k) = d(k) - y(k) \tag{2.42}$$

Let $\xi(k)$ denotes the cost function at iteration k and is given by

$$\xi(k) = \frac{1}{2}\sum_{j=1}^{P} e_j^2(k) \tag{2.43}$$

where P is the number of nodes at the output layer.

The weight vector can be updated by least mean square (LMS) algorithm, as

$$w(k+1) = w(k) - \frac{\mu}{2}\hat{\nabla}(k) \tag{2.44}$$

where $\hat{\nabla}(k)$ is an instantaneous estimate of the gradient of ξ with respect to the weight vector $w(k)$. This gradient is computed as

$$\begin{aligned}\hat{\nabla}(k) &= \frac{\partial \xi}{\partial w} = -2e(k)\frac{\partial y(k)}{\partial w} = -2e(k)\frac{\partial [w(k)s(k)]}{\partial w} \\ &= -2e(k)s(k)\end{aligned} \tag{2.45}$$

Substituting the values of $\hat{\nabla}(k)$ in (2.44) we get

$$w(k+1) = w(k) + \mu e(k)s(k) \tag{2.46}$$

where μ denotes the step-size $(0 \le \mu \le 1)$, which controls the convergence speed of the algorithm.

2.5 Derivative free algorithms / Evolutionary computing based algorithms

2.5.1 Genetic algorithm (GA)

Genetic algorithms are a class of evolutionary computing techniques, which is a rapidly growing area of artificial intelligence. Genetic algorithms are inspired by Darwin's theory of evolution. Simply said, problems are solved by an evolutionary process resulting in a best (fittest) solution (survivor) - in other words, the solution is evolved.

Evolutionary computing was introduced in the 1960s by Rechenberg in his work "*Evolution strategies*" (*Evolutionsstrategie* in original). His idea was then developed by other researchers. Genetic Algorithms (GAs) were invented by John Holland and developed by him and his students and colleagues [2.10]. This led to Holland's book "*Adaption in Natural and Artificial Systems*" published in 1975.

The algorithm begins with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are then selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are, the more chances they have to reproduce.

This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied.

2.5.1.1 Outline of the basic genetic algorithm

1. **[Start]** Generate random population of *n* chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population
3. **[New population]** Create a new population by repeating following steps until the new population is complete
    a **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)

b **[Crossover]** With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.

c **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).

d **[Accepting]** Place new offspring in the new population

4. **[Replace]** Use new generated population for a further run of the algorithm
5. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
6. **[Loop]** Go to step **2**

The outline of the Basic GA provided above is very general. There are many parameters and settings that can be implemented differently in various problems.

Elitism is often used as a method of selection. Which means, that at least one of a generation's best solution is copied without changes to a new population, so the best solution can survive to the succeeding generation.

2.5.1.2 Operators of GA

Overview

The crossover and mutation are the most important parts of the genetic algorithm. The performance is influenced mainly by these two operators.

Encoding of a Chromosome

A chromosome should in some way contain information about solution that it represents. The most commonly used way of encoding is a binary string. A chromosome then could look like this:

Chromosome 1	1101100100110110
Chromosome 2	1101111000011110

Fig. 2.9 Chromosome

Each chromosome is represented by a binary string. Each bit in the string can represent some characteristics of the solution. There are many other ways of encoding. The encoding depends mainly on the problem to be solved. For example, one can encode directly integer or real numbers; sometimes it is useful to encode some permutations and so on.

Crossover

Crossover operates on selected genes from parent chromosomes and creates new offspring. The simplest way how to do that is to choose randomly some crossover point and copy everything before this point from the first parent and then copy everything after the crossover point from the other parent.

Crossover is illustrated in the following Fig. 2.10 (| is the crossover point)

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

Fig. 2.10 Crossover

There are other ways how to make crossover, for example we can choose more crossover points.

Mutation

Mutation is intended to prevent falling of all solutions in the population into a local optimum of the solved problem. Mutation operation randomly changes the offspring resulted from crossover. In case of binary encoding we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Mutation can be then illustrated as follows (Fig. 2.11)

Original offspring 1	1101111000011110
Original offspring 2	1101100100110110
Mutated offspring 1	1100111000011110
Mutated offspring 2	1101101100110110

Fig. 2.11 Mutation

The technique of mutation (as well as crossover) depends mainly on the encoding of chromosomes. For example when we are encoding by permutations, mutation could be performed as an exchange of two genes.

2.5.1.3 Parameters of GA

Crossover and Mutation Probability

There are two basic parameters of GA - crossover probability and mutation probability.

Crossover probability: It indicates how often crossover will be performed. If there is no crossover, offspring are exact copies of parents. If there is crossover, offspring are made from parts of both parent's chromosome. If crossover probability is **100%**, then all offspring are made by crossover. If it is **0%**, whole new generation is made from exact copies of chromosomes from old population (but this does not mean that the new generation is the same!). Crossover is made in hope that new chromosomes will contain good parts of old chromosomes and therefore the new chromosomes will be better. However, it is good to leave some part of old population survives to next generation.

Mutation probability: This signifies how often parts of chromosome will be mutated. If there is no mutation, offspring are generated immediately after crossover (or directly copied) without any change. If mutation is performed, one or more parts of a chromosome are changed. If mutation probability is **100%**, whole chromosome is changed, if it is **0%**, nothing is changed. Mutation generally prevents the GA from falling into local extremes.

Mutation should not occur very often, because then GA will in fact change to **random search**.

Other Parameters

There are also some other parameters of GA. One another particularly important parameter is population size.

Population size: It signifies how many chromosomes are present in population (in one generation). If there are too few chromosomes, then GA has few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, then GA slows down.

2.5.1.4 Selection

Introduction

The chromosomes are selected from the population to be parents for crossover. The problem is how to select these chromosomes. According to Darwin's theory of evolution, the best ones survive to create new offspring. There are many methods in selecting the best chromosomes. Examples are roulette wheel selection, Boltzman selection, tournament selection, rank selection, steady state selection and some others. In this thesis we have used the tournament selection as it performs better than the others.

Tournament Selection

A selection strategy in GA is simply a process that favours the selection of better individuals in the population for the mating pool. There are two important issues in the evolution process of genetic search, population diversity and selective pressure. Population diversity means that the genes from the already discovered good individuals are exploited while promising the new areas of the search space continue to be explored. Selective pressure is the degree to which the better individuals are favoured. The tournament selection strategy provides selective pressure by holding a tournament competition among individuals [2.11].

2.5.1.5 GA for Function optimization

To understand the use of GA, minimization of a multimodal function given in (2.47) is carried out through simulation study.

```
f(k)=...    +5*exp(-0.1*((x-15)^2+(y-20)^2))...
            -2*exp(-0.08*((x-20)^2+(y-15)^2))...
            +3*exp(-0.08*((x-25)^2+(y-10)^2))...
            +2*exp(-0.1*((x-10)^2+(y-10)^2))...
            -2*exp(-0.5*((x-5)^2+(y-10)^2))...
            -4*exp(-0.1*((x-15)^2+(y-5)^2))...
            -2*exp(-0.5*((x-8)^2+(y-25)^2))...
            -2*exp(-0.5*((x-21)^2+(y-25)^2))...
            +2*exp(-0.5*((x-25)^2+(y-16)^2))...
            +2*exp(-0.5*((x-5)^2+(y-14)^2));                    (2.47)
```

The function has global minimum point at [15, 5]. Single point crossover was applied and best 20 individuals having higher fitness values were selected for next generation. Mutation and crossover rate were taken as 0.25 and 0.8 respectively and population size was set at 20. Each parameter was represented by eight bits.

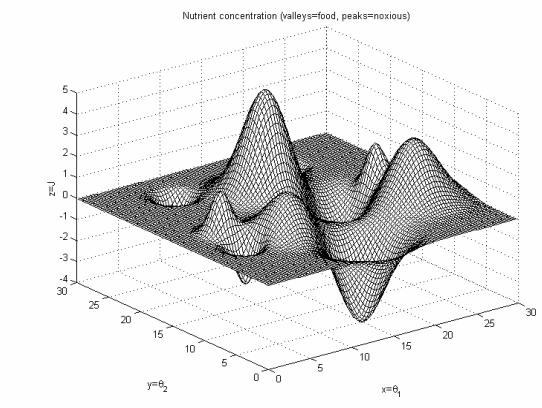


Fig. 2.12 Multimodal function of (2.47)

The fitness value settles to the global minimum value very soon. Results given by GA was [15.1373 5.0980] and $g_{\min}$ = -3.9757.

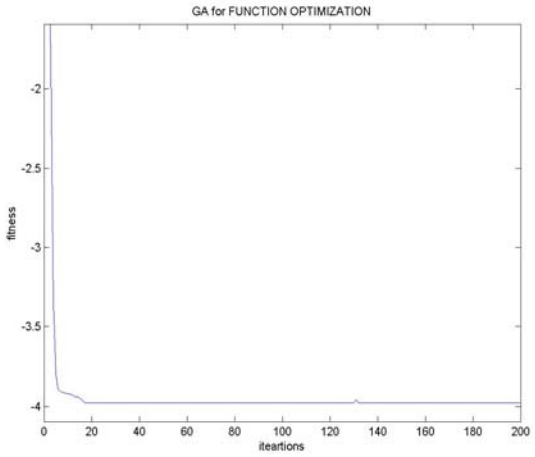


Fig. 2.13 Fitness curve of the function vs iteration

At iteration 1 the result is

Table 2.1

Initial Generation C1

W	Fitness
7.1373 0.3922	-0.0009
-16.2353 -16.3922	0.0000
20.0000 -7.7647	-0.0000
-16.0784 -5.4118	0.0000
7.9216 3.8431	0.0060
9.1765 -10.1176	-0.0000
-13.8824 -8.8627	0.0000
-8.8627 0.5490	0.0000
1.9608 12.4706	0.0069
-15.4510 16.5490	0.0000
2.5882 -14.3529	-0.0000
15.4510 - 16.3922	-0.0000
4.4706 -19.8431	-0.0000
18.5882 -11.2157	-0.0000
-19.6863 -7.6078	0.0000
-10.5882 7.4510	0.0000
13.7255 16.8627	1.5280
15.1373 5.4118	-3.9079
-11.8431 -10.1176	0.0000
19.3725 12.1569	-0.8527

After 400 iterations, all W and fitness values become equal which provides the desired result.

Table 2.2

C1 population after 400th iteration

W	Fitness
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757

2.5.2 Particle swarm optimization (PSO)

2.5.2.1 Basic method

Natural creatures sometimes behave as a swarm. One of the main stream of artificial life researches is to examine how natural creatures behave as a swarm and reconfigure the swarm models inside a computer. Swarm behavior can be modeled with a few simple rules. School of fishes and swarm of birds can be modeled with such simple models.

In 2001 Kennedy and Eberhart developed a PSO [2.12] concept through simulation of bird flocking in two-dimensional space. The position of each agent is represented by XY axes position and also the velocity is expressed by v_x (the velocity of X axis) and v_y (the velocity of Y axis). Modification of the agent position is realized by the position and velocity information. Bird flocking optimizes a certain objective function. Each agent knows its best value so far (pbest) and its XY position. This information is analogy of personal experiences of each agent. Moreover, each agent knows the best value so far in the group

(gbest) among pbests. This information is analogy of knowledge of how the other agents around them have performed. Namely, each agent tries to modify its position using the following informations

1. the current positions (x, y)
2. the current velocities (v_x, v_y)
3. to go to the center of the swarm
4. the distance between the current position and pbest
5. the distance between the current position and gbest

2.5.2.2 Particle swarm optimization algorithm

This modification can be represented by the concept of velocity. Velocity of each agent can be modified by the following equation [2.12] :

$$V_i^{k+1} = wV_i^k + c_1 \times rand_1 \times (pbest_i - s_i^k) + c_2 \times rand_2 \times (gbest_i - s_i^k) \tag{2.48}$$

where

V_i^k : velocity of agent i at iteration k

w : weighting function,

c_j : weighting factor,

$rand$: random number between 0 and 1,

s_i^k : current position of agent i at iteration k

$pbest_i$: personal best of agent i,

$gbest$: global best of the group.

The following weighting function [2.12] is usually utilized in (2.48)

$$w = w_{\max} - \frac{(w_{\max} - w_{\min})}{iter_{\max}} \times iter \tag{2.49}$$

where

$w_{\max}$: initial weight,

$w_{\min}$: final weight,

$iter_{\max}$: maximum iteration number,

$iter$: current iteration number.

Using the above equation, a certain velocity, which gradually gets close to pbest and gbest can be calculated. The current position (searching point in the solution space) can be modified by the following equation [2.12] :

$$S_i^{k+1} = S_i^k + V_i^{k+1} \tag{2.50}$$

The general flow chart of PSO is shown in Fig. 2.14 and the step wise procedure is detailed below

1. **Step. 1** Generation of initial condition of each agent

Initial searching points (s_i^0) and velocities (v_i^0) of each agent are usually generated randomly within the allowable range. The current searching point is set to pbest for each agent. The best-evaluated value of pbest is set to gbest and the agent number with the best value is stored.

2. **Step. 2** Evaluation of searching point of each agent

The objective function value is calculated for each agent. If the value is better than the current pbest of the agent, the pbest value is replaced by the current value. If the best value of pbest is better than the current gbest, gbest is replaced by the best value and the agent number with the best value is stored.

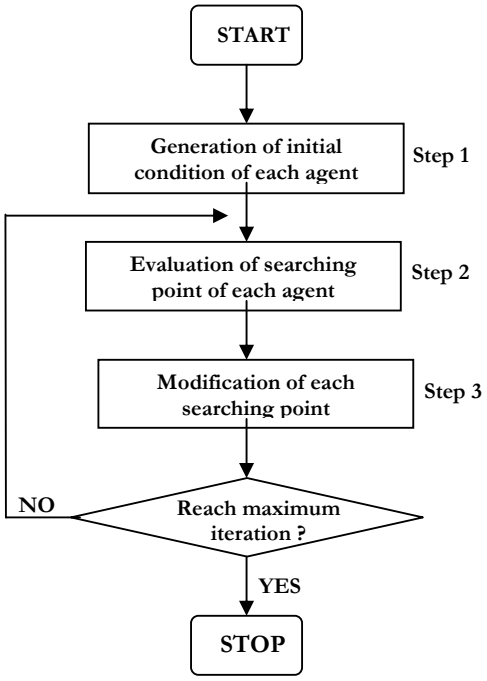


Fig. 2.14 General flow chart of PSO

3. **Step. 3** Modification of each searching point

The current search point of each agent is changed using (2.48), (2.49) and (2.50).

4. **Step. 4** Checking the exit condition

If the current iteration number reaches the predetermined maximum iteration number, then exit. Otherwise, go to step 2.

The features of the searching procedure of PSO can be summarized as follows:

1. As shown in (2.48), (2.49) and (2.50), PSO can essentially handle continuous optimization problem.

2. PSO utilizes several search points like genetic algorithm and the search points gradually get close to the optimal point using their pbest and the gbest information.

3. The first term of the right-hand side (RHS) of (2.48) is corresponding to diversification in the search procedure. The second and third terms of that are corresponding to intensification in the search procedure. This method has a well-balanced mechanism to utilize diversification and intensification in the search procedure efficiently.

4. The PSO can handle continuous optimization problems with continuous state variables in a n-dimension solution space.

Feature (3) can further be explained as follows. The RHS of (2.48) consists of three terms. The first term is the previous velocity of the agent. The second and third terms are utilized to change the velocity of the agent. Without the second and third terms, the agent will keep on "flying" in the same direction until it hits the boundary. Namely, it tries to explore new areas and, therefore, the first term is corresponding to diversification in the search procedure. On the other hand, without the first term, the velocity of the "flying" agent is only determined by using its current position and its best positions in history. Namely, the agents will try to converge to their pbests and/or gbest and, therefore, the terms are corresponding to intensification in the search procedure.

2.5.3 Bacterial foraging optimization (BFO)

2.5.3.1 Introduction

Natural selection tends to eliminate animals with poor "foraging strategies" (methods for locating, handling, and ingesting food) and favor the propagation of genes of those animals that have successful foraging strategies since they are more likely to enjoy reproductive success (they obtain enough food to enable them to reproduce). After

many generations, poor foraging strategies are either eliminated or shaped into good ones (redesigned). Logically, such evolutionary principles have led scientists in the field of "foraging theory" to hypothesize that it is appropriate to model the activity of foraging as an optimization process: A foraging animal takes actions to maximize the energy obtained per unit time spent foraging, in the face of constraints presented by its own physiology and environment

2.5.3.2 Bacterial foraging

Bacteria have the tendency to gather to the nutrient-rich areas by an activity called chemotaxis. It is known that bacteria swim by rotating whip like flagella driven by a reversible motor embedded in the cell wall. E. coli has 8-10 flagella placed randomly on a cell body. When all flagella rotate counterclockwise, they form a compact, helically propelling the cell along a helical trajectory, which is called run. When the flagella rotate clockwise, they pull on the bacterium in different directions, which causes the bacteria to tumble. The four steps involved in bacterial foraging are briefly outlined next.

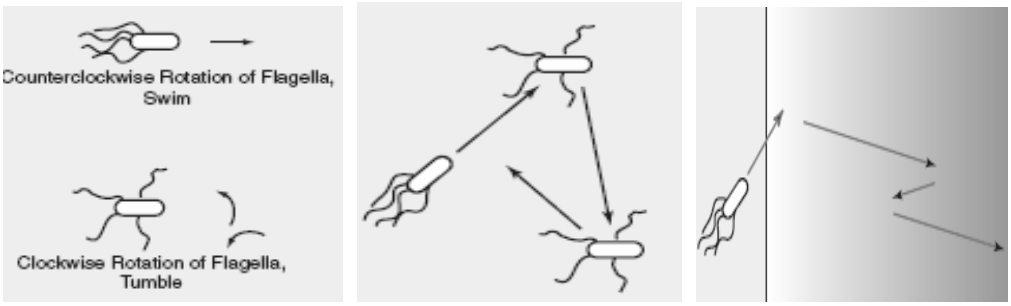


Fig. 2.15 Swimming, Tumbling and Chemotactic behavior of Ecoli

(1)**Chemotaxis** : An E. coli bacterium can move in two different ways; it can run (swim for a period of time) or it can tumble, and alternate between these two modes of operation in the entire lifetime. In the BFO, a unit walk with random direction represnts a tumble and a unit walk with the same direction in the last step indicates a run. After one step move, the position of the i th bacterium can be presented [2.13] as

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\phi(j) \tag{2.51}$$

where $\theta^i(j,k,l)$ represents the i th bacterium at j th chemotactic, k th reproductive and l th elimination and dispersal step. $C(i)$ is the length of unit walk in the random direction. It is assumed to be constant and $\phi(j)$ is the direction angle of the j th step. During run operation, $\phi(j)$ is same as $\phi(j-1)$, otherwise, $\phi(j)$ is a random angle directed within a range of $[0,2\pi]$.

If the cost at $\theta^i(j+1,k,l)$ is better than the cost at $\theta^i(j,k,l)$ then the bactrium takes another step of size $C(i)$ in the same direction otherwise it tumbles. This swim process is as long as it continue to reduce the cost function, but only to a maximum number of steps, N_s.

(2) **Swarming** : The bacteria in times of stresses release attractants to signal bacteria to swarm together. It however also releases a repellant to signal others to be at a minimum distance from it. Thus all of them will have a cell to cell attacrtion via attractant and cell to cell repulsion via repellant. The cell to cell signalling in *E. coli* swarm may be represented [2.13] by the function

$$J_{cc}(\theta,P(j,k,l)) = \sum_{i=1}^{S} J_{cc}(\theta,\theta^i(j,k,l)) = \sum_{i=1}^{S}[-d_a \exp(-w_a \sum_{m=1}^{p}(\theta_m - \theta_m^i)^2)] + \sum_{i=1}^{S} h_r \exp(-w_r \sum_{m=1}^{p}(\theta_m - \theta_m^i)^2)] \quad (2.52)$$

where $J_{cc}(\theta,P(j,k,l))$ represents the objective function value to be added to the actual objective function, S is the total number of bacteria, p is the number of variables to be optimized, $\theta = [\theta_1,\theta_2,.....\theta_p]^T$ is a point in the p -dimensional serach domain, d_a = depth of the attractant, w_a = width of the attractant, h_r = height of the repellant and w_r = width of the repellant.

(3) **Reproduction** : A reproduction step is taken after N_c chemotactic steps. Let $S_r = S_b/2$ be the number of population members who have had sufficient nutrients so that they reproduce i. e. split into two. For reproduction, the population is stored in order of ascending fitness function. The S_r least haelthy bacteria die and the other S_r bacteria

each split into two identical ones which occupy the same positions in the environment. This method keeps the population size constant.

(4) **Elimination and Dispersal** : Since bacteria may be stuck around the initial positions or local optima, it is possible for the diversity of BFA to change either gradually or suddenly to eliminate the possibility of being trapped into local minima. The dispersion vent happens after a certain number of reproduction process. A bacterium is chosen, according to a preset probability p_{ed}, to be dispersed and moved to another position within the environment. These events may prevent the local minima trapping effectively, but unexpectedly disturb the optimization process. The mathematical treatment of this new concept is presented in[2.13].

2.5.4 Artificial immune system (AIS)

The human immune system is a very complex system formed by a large number of cells, molecules and diverse mechanisms. The main function of immunity is to protect our bodies from the invasion of external microorganisms. The cells and molecules responsible for immunity constitute biological immune system (BIS). The AIS is developed by following the principles of BIS. Bersini first used immune algorithms to solve practical problems. The books [2.14, 2.15] provide excellent materials about the various principles and algorithms of AIS. According to BIS theory our body immunity is composed of two defense lines: innate and adaptive immunity. Innate immunity is nonspecific which means that it is independent of the foreign antigen. The adaptive immunity has memory and learning capabilities and it is antigen dependent, meaning that each different type of antigen provokes a different immune response. The main components of the adaptive immunity are the cells called B lymphocytes or simply B cells. When B lymphocytes are stimulated by a specific antigen, they produce a large number of molecules called antibodies, which play a major role in the adaptive immune response.

The clonal selection principle of AIS describes how the immune cells eliminate a foreign antigen and is simple but efficient approximation algorithm for achieving optimum solution. The steps involved in the clonal selection algorithm are

Step 1 : Initialize a number of antibodies(immune cells) which represent initial population size.

Step 2 : When an antigen or pathogen invades the organism; a number of antibodies that recognize these antigens survives. In Fig.2.16, only the antibody C is able to recognize the antigen as its structure fits to a portion of the pathogen. So fitness of antibody C is higher than others.

Step 3 : The immune cells recognize antigens undergo cellular reproduction. During reproduction the somatic cells reproduce in an asexual form, i.e. there is no crossover of genetic material during cell mitosis. The new cells are copies (clones) of their parents as shown for antibody C.

Step 4 : A portion of cloned cells undergo a mutation mechanism which is known as somatic hypermutation as described in [2.15] .

Step 5 : The affinity of every cell with each other is a measure of similarity between them. It is calculated by the distance between the two cells. The antibodies present in a memory response have on average a higher affinity than those of early primary response. This phenomenon is referred to as maturation of immune response. During the mutation process the fitness as well as the affinity of the antibodies gets changed. In each iteration after cloning and mutation those antibodies which have higher fitness and higher affinity are allowed to enter the pool of efficient cells. Those cells with low affinity or self-reactive receptors must be efficiently eliminated.

Step6: At each iteration among the efficient immune cells some become effecter cells (Plasma Cell), while others are maintained as memory cells. The effecter cells secrete antibodies and memory cells having longer span of life so as to act faster or more effectively in future when the organism is exposed to same or similar pathogen.

Step7: The process continues till the termination condition is satisfied else steps 2 to 7 are repeated.

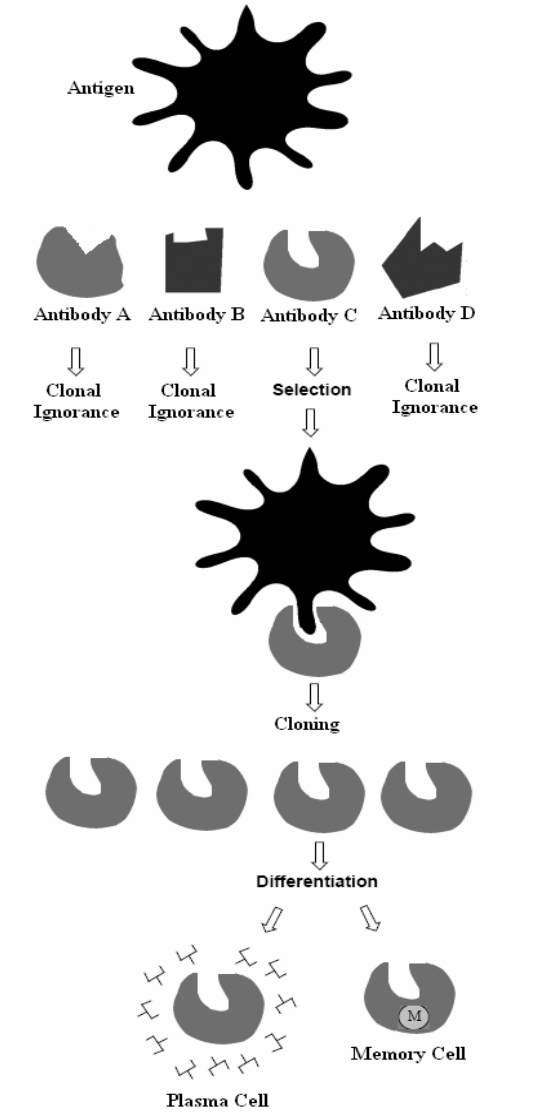


Fig. 2.16 The Clonal Selection Principle

References

[2.1] B. Widrow and S.D. Sterns, "*Adaptive Signal Processing"* Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1985.

[2.2] S. Haykin, "*Adaptive Filter Theory*", 4th edition, Pearson Education Asia, 2002.

[2.3] S. Haykin, "*Neural Networks: A comprehensive foundation*" 2nd Edition, Pearson Education Asia, 2002.

[2.4] E. J. Dayhoff, "*Neural Network Architecture – An Introduction*" Van Norstand Reilold, New York, 1990.

[2.5] N. K. Bose and P. Liang., "*Neural Network Fundamentals with Graphs, Algorithms, Applications*", TMH Publishing Company Ltd, 1998.

[2.6] Richard O. Duda, Peter E. Hart and David G. Stork, "*Pattern Classification*", 2nd edition, John Wiley & Sons, INC.2001.

[2.7] Y.H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison Wesley, Reading, Massachusetts, 1989.

[2.8] Y. H. Pao, G. H. Park and D. J. Sobjic., "Learning and Generalization Characteristics of the Random Vector Function", Neuro Computation, 6: 163-180, 1994.

[2.9] J. C. Patra and R. N. Pal, "A Functional Link Artificial Neural Network for Adaptive Channel Equalization", Signal Processing, vol.43, no.2, pp.181-195, May 1995.

[2.10] D.E.Goldberg, "*Genetic algorithms in search, optimization and machine learning*", Addition-Wesley,1989.

[2.11] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in GA", *Foundations of genetic Algorithms*, I, pp. 53-69, 1991.

[2.12] J. Kennedy, R. C. Eberhart and Y. Shi, "*Swarm intelligence",* San Francisco: Morgan Kaufmann Publishers, 2001.

[2.13] K. M. Passino, "Biomimicry of Bacterial Foraging for distributed optimization and control", IEEE control system magazine, vol 22, issue 3, pp. 52-67, June 2002.

[2.14] D.Dasgupta, *Artificial Immune Systems and their Applications*, Springer-Verlag, 1999.

[2.15] L N de Castro and F. J. Von Zuben , "Learning and Optimization using Clonal Selection Principle," IEEE Trans on Evolutionary Computation, Special issue on Artificial Immune Systems, vol. 6, issue 3, pp.239-251, 2002.

Chapter

3

Development of a New Cascaded Functional Link Artificial Neural Network (CFLANN) for Nonlinear Dynamic System Identification

3.1 Introduction

THE identification of nonlinear dynamic system plays a key role in control, communication and pattern recognition areas [3.1]. It finds wide applications in many diverse fields such as electronic circuit design, environmental system analysis, biological and medical systems and control system design [3.2]. The dynamic system identification task is basically a model estimation process of capturing the dynamics of the system using the measured data. This adaptive model can be used for prediction, system design as well as control. Because of the emerging importance of identification problems various attempts are currently underway to develop efficient nonlinear dynamic models of

complex real processes [3.38].

In recent years, the artificial neural network (ANN) has proved to be a potential tool for system identification in a highly nonlinear dynamic environment. The ANN model has two distinct advantages : (i) inherent ability to learn by optimizing an appropriate error function (ii) excellent approximation capability to match a nonlinear function. The ANN architectures used for this purpose are (i) the multilayer artificial neural network (MLANN), (ii) the radial basis function (RBF) network, (iii) the recurrent neural network (RNN) and (iv) the functional link ANN (FLANN). The MLANN trained with back propagation (BP) algorithm and many variations of it have been successfully employed for system identification task [3.3] - [3.7]. The MLANN structure offers robustness and effective modeling and control capability of complex dynamic plants. Several real processes such as control of truck-backer-upper problem [3.5] and robot arm [3.6] have also employed the MLANN. However this is associated with some inherent drawbacks, e.g., multiple local minima problem, the difficulty of selecting the number of hidden units and the possibility of over fitting.

Subsequently the RBF neural networks have been introduced which have proved to be a useful alternative to the MLANN for effective identification of nonlinear dynamic systems [3.8]-[3.9]. The RBF neural network employs a simple structure, can learn functions with local variations and discontinuities effectively and also possesses good universal approximation capability [3.10].However, the RBF has the drawback that both the number and location of its centers must be chosen appropriately. In recent years wavelet neural network (WNN) [3.11], employing nonlinear wavelet basis functions, have been proposed as useful approaches to nonlinear system identification [3.12]-[3.16]. The use of $\beta-$spline and neuro fuzzy functions in place of wavelet basis function have also been suggested for system identification [3.16]. The advantage of using WNN is that the local characteristics of the wavelets enables efficient estimation of regressive functions resulting in localized regularities. Various forms of RNN and recurrent neuro-fuzzy model have also been successfully employed for identification and control tasks [3.17]-[3.22]. These networks are essentially dynamic systems where the internal states evolve according to certain nonlinear state equations. Because of their dynamic nature the identification of complex dynamic systems has been successfully validated. However this approach does not provide any solution to obtain minimal dimensions of the associated recurrent structure.

The learning capabilities of various neural networks used for system identification have significant effects on the overall performance of the adaptive systems. If the information content of the data input to the network can be modified in a suitable manner, the network would able to extract the hidden features of the data. This is the prime motivation behind functional link mapping used in FLANN. The functional link expansions map the original input space into higher dimensions which help to reduce the burden during the training phase of the networks. The functional link acts on each element of the input vector and generates a set of linearly independent functions. They represent the enhanced version of the input information and thus helps to improve the training time and learning accuracy. The FLANN is a single neuron single layer network first proposed by Pao [3.23]. The structure of the FLANN is simple as it represents a flat net with no hidden layers. Therefore the computation and learning algorithm used in the architecture is straight forward. It has already been applied to many diverse fields which include function approximations[3.25]-[3.26], pattern classification [3.23]-[3.24], intelligent control [3.27], nonlinear channel equalization [3.28]-[3.29], system identification [3.30]-[3.32], heating, ventilating and air conditioning (HVAC) system [3.33], signal enhancement [3.34], nonlinearity compensation of sensors [3.35]-[3.36] and financial forecasting [3.37]. It is in general observed that in many of these applications the number of functional expansions involved in the FLANN structure is exceedingly large. This leads to an increase in the computational burdens, hence the overall learning time. Thus there is a need to introduce an alternative ANN structure for modeling the nonlinear plants which associates structural simplicity, less computational load and performance equivalent to or better than those offered by either the MLANN or FLANN based models.

Therefore the main objective of the current work is to introduce an efficient new FLANN model known as cascaded FLANN (CFLANN) with its learning algorithm for nonlinear dynamic system identification task and evaluate its performance by conducting simulation based identification experiments of typical nonlinear plants. The performance of the proposed model is validated and compared by using the same identification examples in MLANN and FLANN based approaches.

3.2 Nonlinear dynamic system identification

System identification is a key problem in system theory and is related with the mathematical representation of a system. A model of a system is represented by an operator P from an input space X into an output space Y and the objective is to characterize the class P to which P belongs. The problem of identification is to obtain a class $\hat{P} \subset P$ and an element $\hat{\mathrm{P}} \in \hat{P}$ so that $\hat{\mathrm{P}}$ approximates P in some predefined sense. In static systems X and Y represent subsets of R^n and R^m respectively, but in dynamical systems these are assumed to be bounded Lebesgue integrable function in the interval $[0,T]$ or $[0,\infty]$ [3.4]. The choice of the class of identification models and the specific methods used to determine $\hat{P}$ is related to the desired accuracy and the analytical tractability. In many practical situations these decisions depend upon prior information associated with the plant to be identified.
Fig. 3.1 shows the identification scheme of a dynamic nonlinear system.

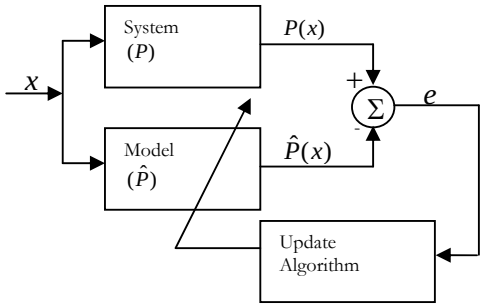


Fig. 3.1 Identification scheme of a dynamic system

In this method compact sets $\mathrm{X}_i \subset R^n$ are mapped into elements $y_i \in R^m; (i = 1, 2, ...)$ in the output space by the operator P. The elements of X_i denote the input patterns corresponding to class y_i. In dynamical systems the operator P of a given plant is defined by the input-output pairs $\{x(t), y(t)\},\ t \in [0,T]$. The objective of identification is to obtain $\hat{\mathrm{P}}$ so that

$$\left\|\hat{y}-y\right\| = \left\|\hat{\mathrm{P}}(x)-\mathrm{P}(x)\right\| \le \varepsilon, x \in \mathrm{X} \tag{3.1}$$

where ε is a pre-specified small value and $\|.\|$ denotes the norm on the output space. In Fig. 3.1 $\hat{\mathrm{P}}(x)=\hat{y}, \mathrm{P}(x)=y$ and e denote the output of the model, the output of the system and the error between the two respectively. Therefore e is given by

$$e = y - \hat{y} \tag{3.2}$$

The input x is assumed to be zero mean uniformly distributed random values lying between -1 to +1. The stability of the plant is assumed with a known parameterization but with unknown parameter values. In the present study single-input single-output (SISO) plants of four different nonlinear models are considered. These plants are described by the nonlinear difference equations (3.3)-(3.6) given in [3.4].

Model 1 :

$$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + g\left[x(k), x(k-1), \ldots\ldots, x(k-m+1)\right] \tag{3.3}$$

Model 2 :

$$y(k+1) = f[y(k), y(k-1), \ldots\ldots\ldots, y(k-n+1)] + \sum_{i=0}^{m-1} \beta_i x(k-i) \tag{3.4}$$

Model 3 :

$$y(k+1) = f[y(k), y(k-1), \ldots\ldots\ldots, y(k-n+1)] + g[x(k), x(k-1), \ldots\ldots\ldots\ldots, x(k-m+1)] \tag{3.5}$$

Model 4 :

$$y(k+1) = f[y(k), y(k-1), \ldots\ldots\ldots, y(k-n+1) + x(k), x(k-1), \ldots\ldots\ldots\ldots, x(k-m+1)] \tag{3.6}$$

In these models $x(k)$ and $y(k)$ represent the input and the output of the SISO plant at the k th time instant respectively and $m \le n$. α_i and β_i are coefficients of the linear combiner part of the models. In this chapter the nonlinear part of the given plants, $f(.)$ and $g(.)$ are modeled by Cascaded FLANN (CFLANN) structures instead of using the MLANN structure as suggested in [3.4]. It is assumed that the plants under consideration are bounded-input-bounded-output (BIBO) stable. In order to achieve stability and to ensure that the parameters of the model converge, a series-parallel scheme [3.4] is employed. In this scheme the output of the plant instead of that of the ANN models is fed back to the models during the training operation.

3.3 Cascaded functional link artificial neural network

Before dealing with the CFLANN adaptive structure and its training algorithm this section first introduces the FLANN model and its associated training rules.

3.3.1 The functional link artificial neural network (FLANN)

When the FLANN structure is employed for identification and control task, its learning capability significantly affects its performance. In this model the input is nonlinearly mapped so that the network extracts the associated hidden information of the data. This is the main motivation behind functional link mapping [3.4] of the FLANN structure. Such mapping also reduces the need of addition layers of the network. The block diagram of an FLANN structure is shown in Fig. 3.2. Let the input signal vector be represented as

$$\underline{X}(k)=[x(k)\,x(k-1).....x(k-m+1)]^T \tag{3.7}$$

Then the functional expansion (FE) block maps each element $x(k)$ into $(2p+1)$ nonlinearly expanded independent components. For trigonometric expansion $\underline{\phi}[x(k)]$ is given by

$$\begin{aligned}\underline{\phi}[x(k)] &= [x(k), \cos\{\pi x(k)\}, \sin\{\pi x(k)\}.... \cos\{p\pi x(k)\}, \sin\{p\pi x(k)\}]^T \\ &= [\phi_1\{x(k)\}, \phi_2\{x(k)\},.... \phi_{2p+1}\{x(k)\}]^T\end{aligned} \tag{3.8}$$

where p is an integer. When each element of $\underline{X}(k)$ is expanded then the expanded vector is represented as

$$\underline{\phi}[\underline{X}(k)]=[\phi_1\{x(k)\}....\phi_{2p+1}\{x(k)\}\phi_{2p+2}\{x(k-1)\}...\phi_{2(2p+1)}\{x(k-1)\}....\phi_N\{x(k-m)\}]^T \tag{3.9}$$

where $N=m(2p+1)$ and m is the number of signal samples fed into the FLANN model. The output $\hat{y}(k)$ of Fig. 3.2 is then given by

$$\hat{y}(k)=\underline{f}\{\underline{\phi}^T(\underline{X}(k))\underline{W}(k)+b(k)\} \tag{3.10}$$

where $b(k)$ is the bias weight and $f\{.\}$ denotes the tanh function. $\underline{W}(k)$ represents the weight vector and is defined as

$$\underline{W}(k)=[w_1(k), w_2(k).....w_N(k)]^T \tag{3.11}$$

The weights of the FLANN model are trained using the algorithm

$$\underline{W}(k+1)=W(k)+\mu\underline{\phi}\{\underline{X}(k)\}.e(k)(1-\hat{y}^2(k)\ ) \tag{3.12}$$

where the error term

$$e(k) = y(k) - \hat{y}(k) \tag{3.13}$$

The convergence coefficient is represented by μ and its value lies between 0 and 1. Equations (3.8), (3.9), (3.10) and (3.12) represent the key equations of FLANN algorithm.

3.3.2 Cascaded functional link artificial neural network (CFLANN)

For the identification of complex nonlinear static and dynamic systems the number of branches in the FLANN increases exponentially with the increase in the complexity of the identification problem. As a result the structural complexity increases in case of FLANN model and even then at times the performance is not improved. Keeping this in view a structurally simple model known as a two-stage FLANN model is proposed which is expected to offer less computational load. In this case the output of the first FLANN stage undergoes another functional expansion. The weights of cascaded FLANN of both the stages are updated by using a CFLANN algorithm. Referring to the proposed CFLANN identification model in Fig. 3.3, the output of stage-1 is given by

$$\hat{y}_2(k) = \tanh\{\hat{y}_1(k)\} \tag{3.14}$$

where $\hat{y}_1(k)$ is given by (3.10).

In the second stage the output $\hat{y}_2(k)$ undergoes nonlinear expansions in FE2 block. Its output is represented as

$$\underline{\psi}\{\hat{y}_2(k)\} = [\psi_1\{\hat{y}_2(k)\}\,\psi_2\{\hat{y}_2(k).....\psi_M\{\hat{y}_2(k)\}]^T \tag{3.15}$$

where $\psi_i\{\hat{y}_2(k)\}$ denotes the i th trigonometric expansion of $\{\hat{y}_2(k)\}, 1 < i < M$. The estimated output of the second stage i. e. the output of the CFLANN model is then given by

$$\hat{y}(k) = \tanh\{\hat{y}_3(k)\} \tag{3.16}$$

where

$$\hat{y}_3(k) = \underline{\psi}^T\{\hat{y}_2(k)\}\underline{H}(k) + b_2(k) \tag{3.17}$$

and

$$\underline{H}(k) = [h_1(k), h_2(k), ..., h_M(k)]^T \tag{3.18}$$

The weights of the second stage are trained using (3.19)

$$\underline{H}(k+1) = \underline{H}(k) + \mu\underline{\psi}\{\hat{y}_2(k)\}e(k)(1-\hat{y}^2(k)) \tag{3.19}$$

where

$$e(k) = y(k) - \hat{y}(k) \tag{3.20}$$

and $y(k)$ is the output of the plant or system to be identified. Based on the principle of back propagation algorithm, a weight update algorithm for CFLANN model is derived. The corresponding weight-update equation of the first stage is derived and is given by

$$\underline{W}(k+1) = \underline{W}(k) + \mu e(k)(1-\hat{y}(k))(1-\hat{y}_2{}^2(k))[\underline{H}^T(k+1)\underline{\psi}'\{\hat{y}_2(k)\}]\underline{\phi}\{\underline{X}(k)\} \quad (3.21)$$

where $\underline{\psi}'\{\hat{y}_2(k)\}$ is the first order derivative of $\underline{\psi}\{\hat{y}_2(k)\}$.

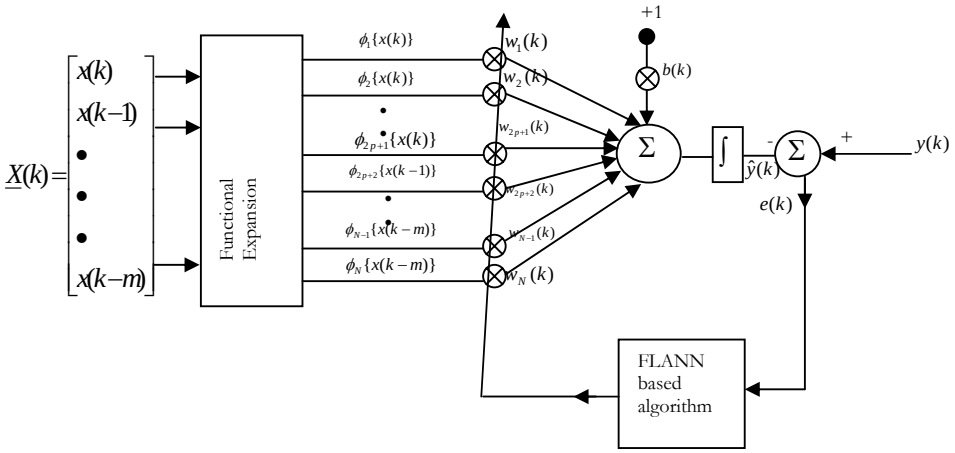


Fig. 3.2 A FLANN model for identification of nonlinear dynamic systems

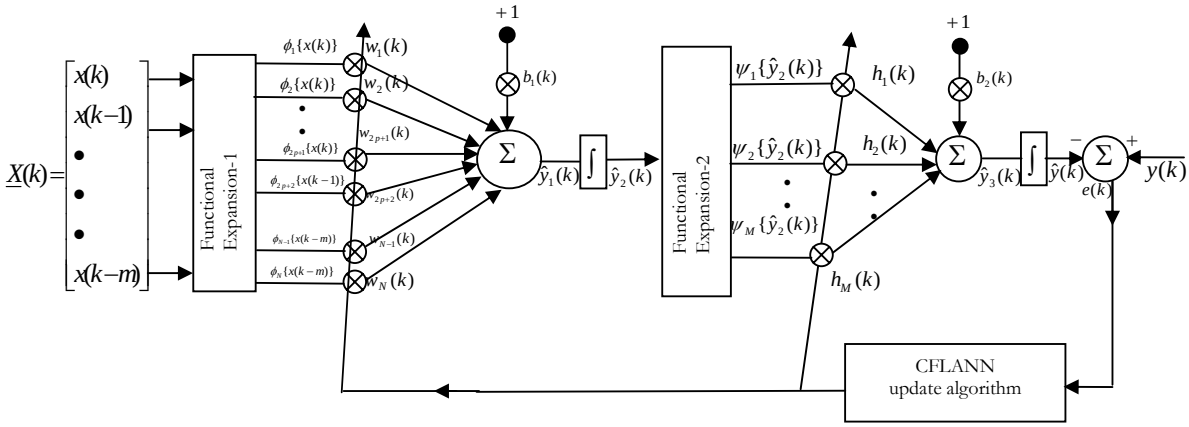


Fig. 3.3 A CFLANN model for identification of nonlinear dynamic systems

Equations (3.10), (3.14), (3.16), (3.17), (3.19) and (3.21) represent the key equations relating to the CFLANN algorithm.

3.4 Simulation study

To demonstrate the performance of the proposed CFLANN model, simulation results of nonlinear identification of four typical dynamic plants are presented. In these examples, the series-parallel model is used to identify the given dynamic plants and CFLANN algorithm is used to adjust the connecting weights of the CFLANN structure. To compare the performance of the proposed models the same examples are also simulated using MLANN and FLANN models. Keeping the best possible performance as the basis the parameters of FLANN and CFLANN were adjusted suitably. For training the MLANN and FLANN models 50, 000 iterations are carried out by using an uniformly distributed random signal over the interval [-1,1] as input. During the testing phase, the effectiveness of the proposed models is studied by using the parallel scheme where the input to the identified model [3.4] used is

$$x(k)=\begin{cases}\sin\dfrac{2\pi k}{250} \ for\ k\le 250\\ 0.8\sin\dfrac{2\pi k}{250}+0.2\sin\dfrac{2\pi k}{25} \ for\ k>250\end{cases} \tag{3.22}$$

Example 1: The difference equation of the nonlinear plant [3.4] to be identified is given as

$$y(k+1)=0.3y(k)+0.6y(k-1)+g[x(k)] \tag{3.23}$$

The linear parameters are 0.3 and 0.6 and the unknown function $g(.)$ is considered as one of the nonlinear functions defined in (3.24)-(3.26).

$$g\ (x)=\frac{4.0x^3-1.2x^2-3.0x+1.2}{0.4x^5+0.8x^4-1.2x^3+0.2x^2-3.0} \tag{3.24}$$

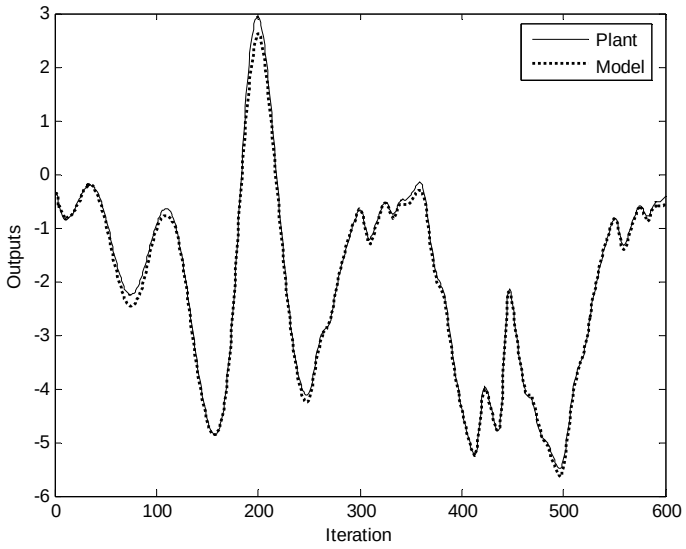
$$g(x)=0.5\sin^3(\pi x)-\frac{2.0}{x^3+2.0}-0.1\cos(4\pi x)+1.125 \tag{3.25}$$

$$g(x)=0.6\sin(\pi x)+0.3\sin(3\pi x)+0.1\sin(5\pi x) \tag{3.26}$$

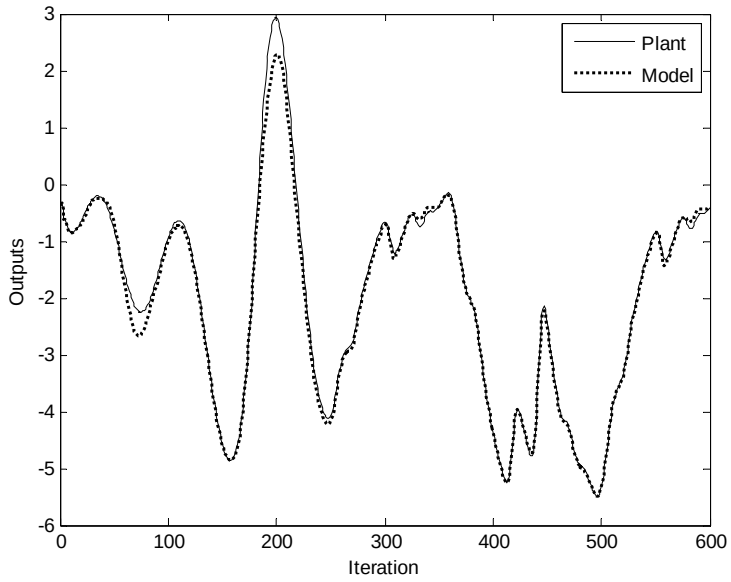
To identify the plant a series-parallel model described by the difference equation (3.27) is used

$$\hat{y}(k+1)=0.3y(k)+0.6y(k-1)+\mathrm{N}[x(k)] \tag{3.27}$$

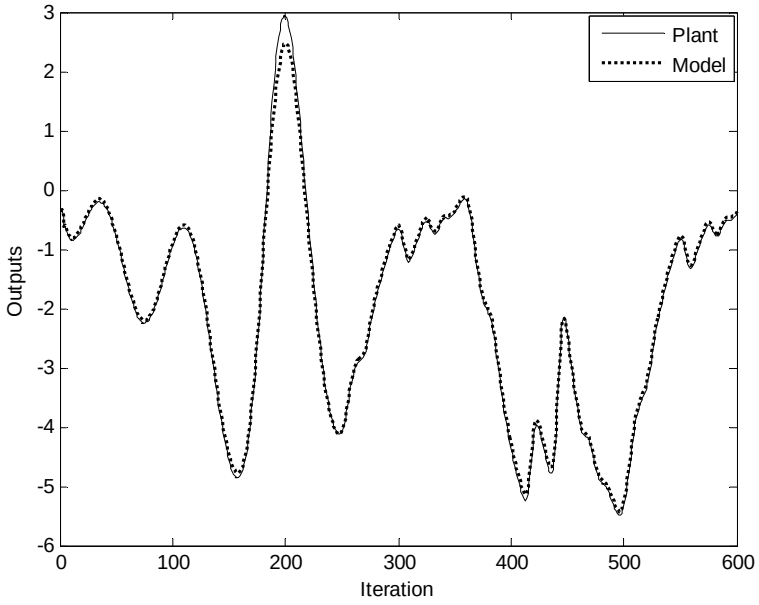
where $N[x(k)]$ represents one of the MLANN, FLANN and CFLANN models. The MLANN model used for this purpose has {1-20-10-1} structure. The FLANN input is expanded to 14 terms by using trigonometric expansion. Both the convergence parameter, μ and the momentum parameter, η are chosen to be 0.1 for both the models. But in case of CFLANN model the input is expanded into 10 terms (5 terms in first stage and 5 terms in second stage) in examples shown in (3.24), 12 stages (7 in first stage and 5 in second stage) in examples shown in (3.25) and (3.26). The coefficient μ is chosen to be 0.1 in all examples. The comparison of responses of these dynamic systems are provided in Figs. 3.4(a)-3.4(i). From all these plots of Fig. 3.4, it is in general observed that the identification performance of CFLANN model is better than those obtained from the MLANN and FLANN models. The sum of squared errors (SSE) computed between the actual and estimated responses of various methods are shown in Table 3.1. It also shows improved performance of the proposed method compared to those offered by the other two.



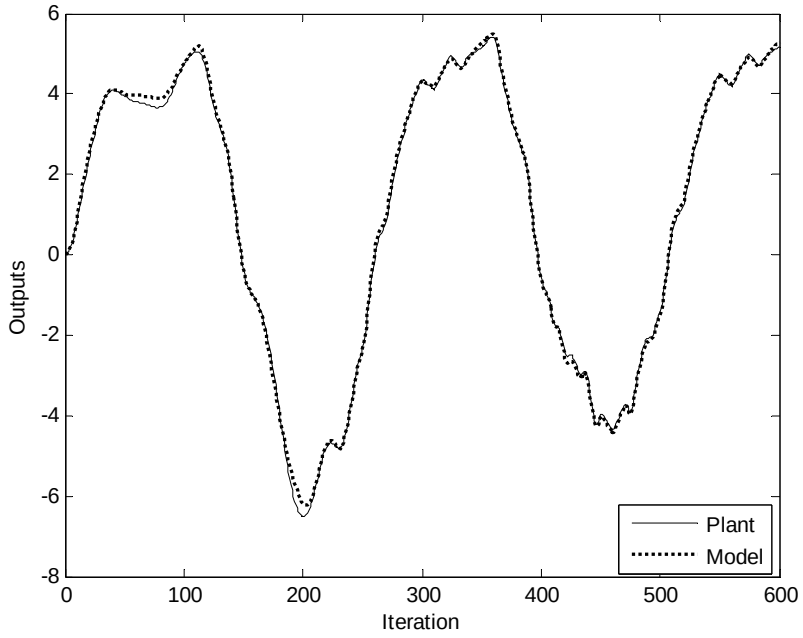
(a) Comparison of output response using CFLANN method (Example 1 with nonlinearity in (3.24))



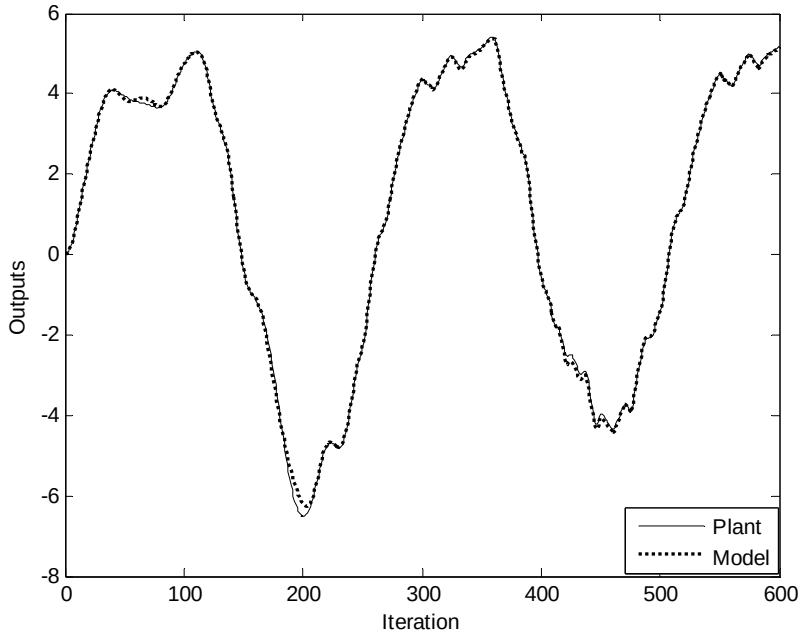
(b) Comparison of output response using FLANN method (Example 1 with nonlinearity in (3.24))



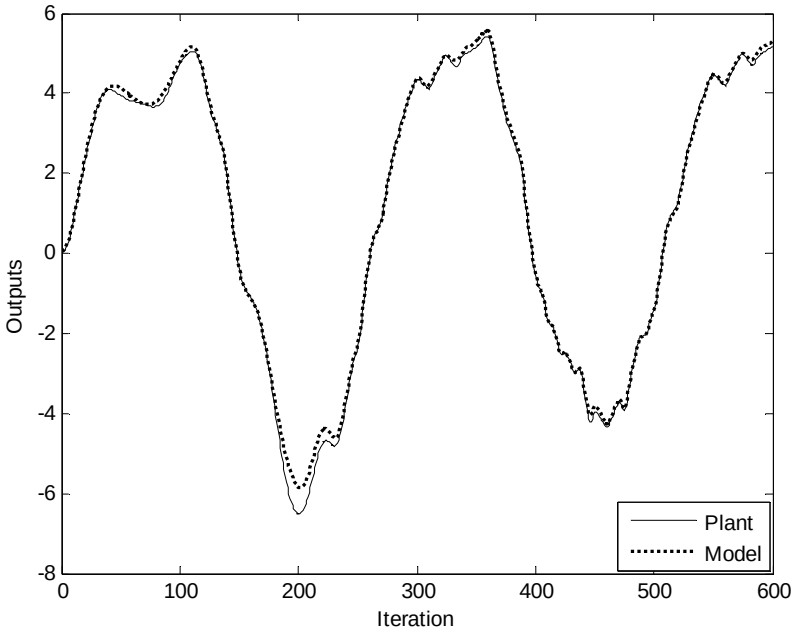
(c) Comparison of output response using MLANN method (Example 1 with nonlinearity in (3.24))



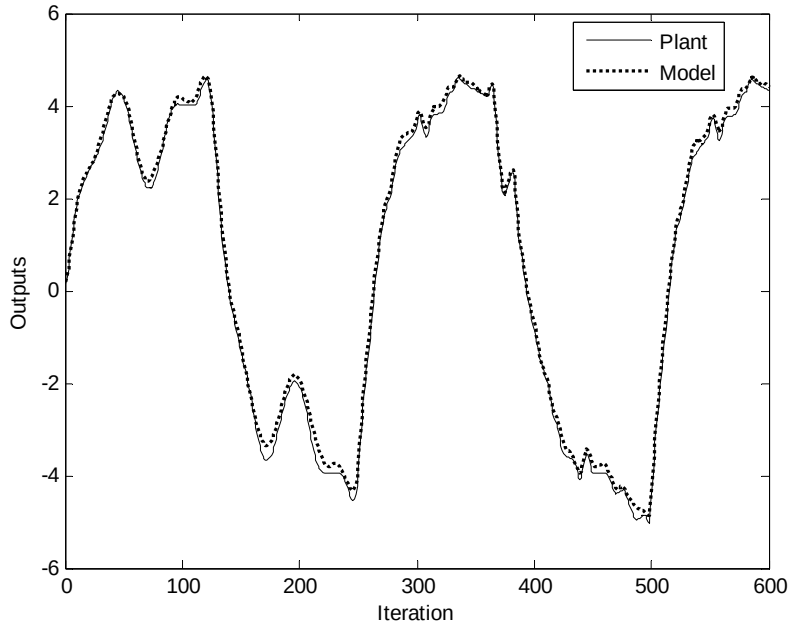
(d) Comparison of output response using CFLANN method (Example 1 with nonlinearity in (3.25))



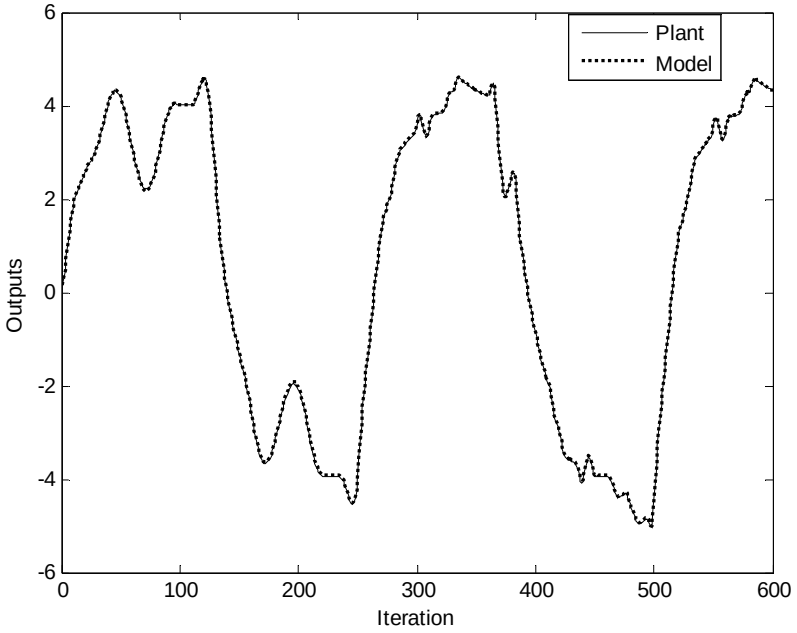
(e) Comparison of output response using FLANN method (Example 1 with nonlinearity in (3.25))



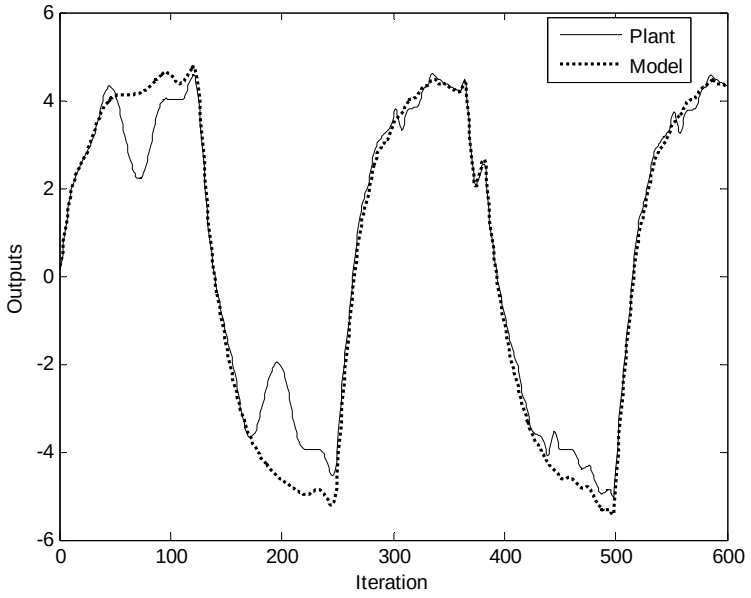
(f) Comparison of output response using MLANN method (Example 1 with nonlinearity in (3.25))



(g) Comparison of output response using CFLANN method (Example 1 with nonlinearity in (3.26))



(h) Comparison of output response using FLANN method (Example 1 with nonlinearity in (3.26))



(i) Comparison of output response using MLANN method (Example 1 with nonlinearity in (3.26)

Fig. 3.4 Comparison of identification performance of nonlinear plants of (Example -1)

Example 2: In this example [3.4] the plant to be identified is of type Model-2 and is represented by the difference equation

$$y(k+1) = f[y(k), y(k-1)] + x(k) \tag{3.28}$$

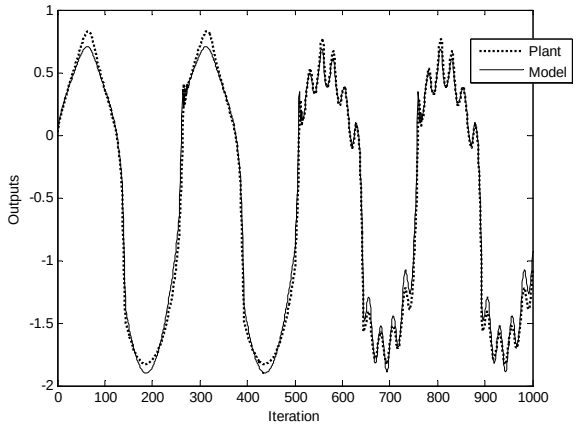
The unknown nonlinear function f is given by

$$f(y_1, y_2) = \frac{y_1 y_2 (y_1 + 2.5)(y_1 - 1.0)}{1.0 + y_1^2 + y_2^2} \tag{3.29}$$

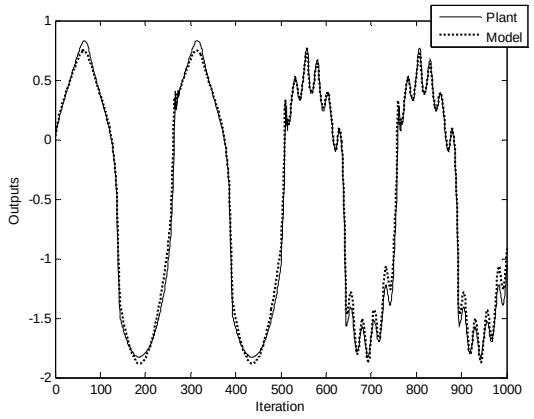
In this case the series-parallel scheme used to identify the plant is given as

$$\hat{y}(k+1) = \mathrm{N}[(y(k), y(k-1)] + x(k) \tag{3.30}$$

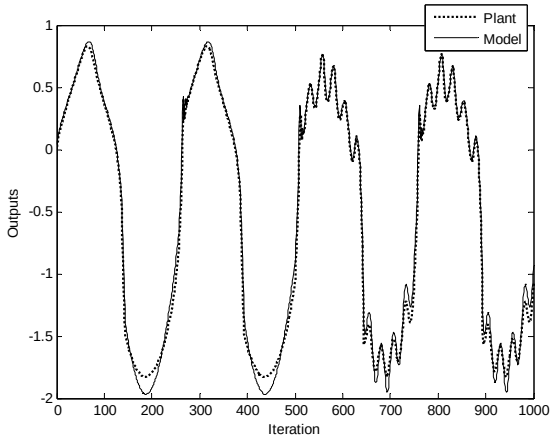
In the simulation study the structure used for MLANN is {2-20-10-1}. In FLANN, the two inputs are expanded into 24 terms and the values of convergence parameter, μ and momentum parameter, η are set at 0.05 and 0.1 respectively in both MLANN and FLANN models. In case of the CFLANN model the two dimensional input is expanded into 17 terms (14 term in the first stage and 3 terms in the second stage). The value of μ is chosen to be 0.1. The response obtained from the plant and various models are compared in Figs. 3.5(a)-(c). In this case also it is observed that the identification performance of the CFLANN is better than those obtained from the other two. It is also observed from the magnitude of the SSE shown in Table 3.1.



(a) Comparison of output response using CFLANN method



(b) Comparison of output response using FLANN method



(c) Comparison of output response using MLANN method

Fig. 3.5 Comparison of identification performance of nonlinear plant of Example-2

Example 3: In this case the plant [3.4] belongs to Model-3 type and is given by the difference equation

$$y(k+1) = f[y(k) + g(x(k)] \tag{3.31}$$

where the unknown nonlinear functions $f(.)$ and $g(.)$ are represented as

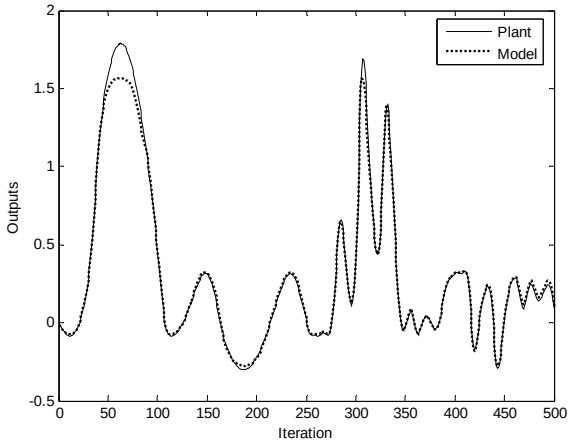
$$f(y) = \frac{y(y+0.3)}{1.0+y^2} \tag{3.32}$$

$$g(x) = x(x+0.8)(x-0.5) \tag{3.33}$$

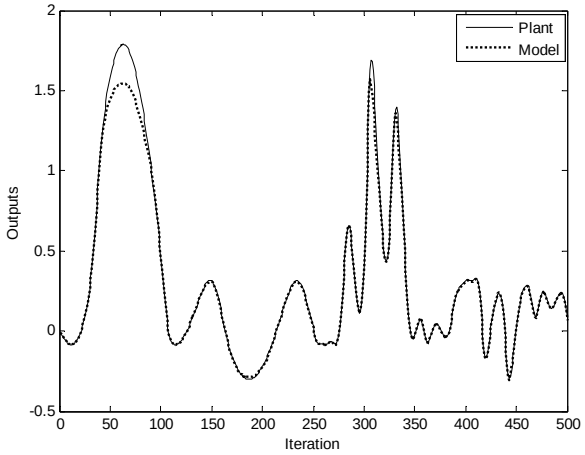
The series-parallel scheme used is given by

$$\hat{y}(k+1) = \mathrm{N}_1[y(k)] + \mathrm{N}_2[x(k)] \tag{3.34}$$

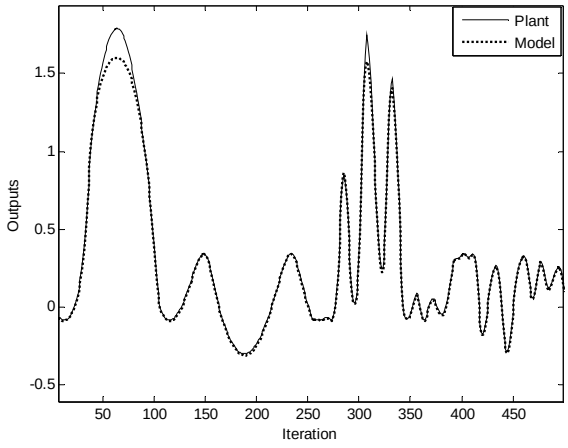
where N_1 and N_2 represent one of the MLANN, FLANN or CFLANN model. In MLANN the structure used for both N_1 and N_2 are having the structure {1-20-10-1}, whereas in case of FLANN, 14 and 24 terms trigonometric expansions are used. The values of convergence parameter, μ and momentum parameter, η are chosen to be 0.1 in both MLANN and FLANN models. In the CFLANN model each of the N_1 and N_2 is expanded into 14 terms (7 in first stage and 7 in second stage) and μ is taken as 0.1. The results of identification are plotted in Figs. 3.6(a)-(c). It may be seen that the CFLANN model is estimating the plant response better than that of the MLANN and FLANN based methods. The SSE of Table 3.1 also indicates the same trend.



(a) Comparison of output response Using CFLANN method



(b) Comparison of output response using FLANN method



(c) Comparison of output response using MLANN

Fig. 3.6 Comparison of identification performance of nonlinear plant of Example-3

Example 4: The plant [3.4] in this case belongs to Model-4 and is described by the difference equation

$$y(k+1) = f[y(k), y(k-1), y(k-2), x(k), x(k-1)] \tag{3.35}$$

where the unknown nonlinear function f is given by

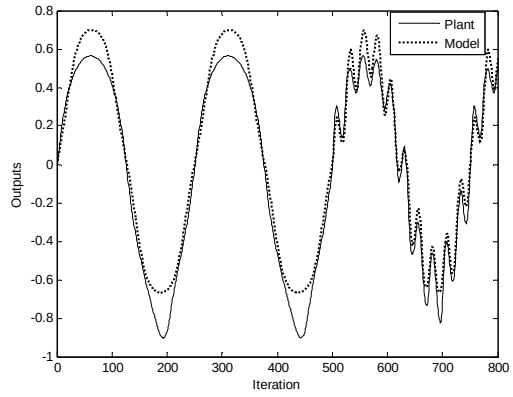
$$f[a_1, a_2, a_3, a_4, a_5] = \frac{a_1 a_2 a_3 a_5 (a_3 - 1.0) + a_4}{1.0 + a_2^{\,2} + a_3^{\,2}} \tag{3.36}$$

The series-parallel model used for identification of this plant is given as

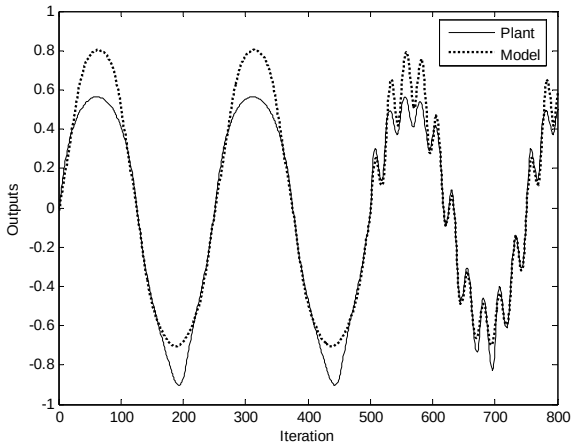
$$\hat{y}(k+1) = \mathrm{N}[y(k), y(k-1), y(k-2), u(k), u(k-1)] \tag{3.37}$$

In the MLANN model N represents a {5-20-10-1} structure and in case of FLANN model the input and output are expanded to ten terms each using trigonometric expansion. Both the parameters μ and η are chosen to be 0.1 in these two cases.

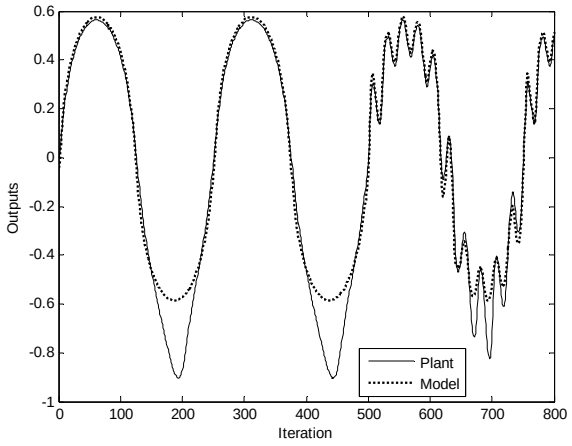
In case of CFLANN model the input is expanded to 4 terms and output is expanded to 6 terms in the first stage and in the second stage the output is expanded to 3 terms. The value of μ chosen is 0.1. Figs. 3.7(a)-(c) show the comparative performance of the output response of two models. The simulation results also indicate that the identification performance is best in the proposed model as may be evident from comparison of SSE shown in Table 3.1.



(a) Comparison of output response using CFLANN model



(b) Comparison of output response using FLANN model



(c) Comparison of output response using MLANN model

Fig. 3.7 Comparison of identification performance of nonlinear plant of Example –4

The SSE computed by comparing the desired and the estimated outputs of the four identification examples of three different methods is shown in Table 3.1. It is observed that the proposed model yields minimum SSE in all cases compared to that obtained from the other two techniques.

Table 3.1

Comparison of the sum of squared errors (SSE) between the plant and the model outputs

Nonlinear plants	SSE		
	CFLANN	FLANN	MLANN
Ex-1 using (3.24)	5.52	5.76	7.74
Ex-1 using (3.25)	3.84	4.02	16.68
Ex-1 using (3.26)	0.1917	0.1846	291.18
Example-2	5.40	5.40	5.70
Example-3	1.15	1.70	1.20
Example-4	6.88	7.92	7.36

The computational complexity required in the identification of the plants given in examples 1-4 is presented in Table 3.2. It is in general observed that the proposed CFLANN model involves the lowest computational complexity compared to other two models. Further it is in general observed that the CFLANN offers best identification performance compared to the existing MLANN and FLANN based models.

Table 3. 2

Comparison of Computational Complexity of various system identification models

Types of models	No. of tanh ( )	No. of Cos/Sin	No. of weights	No. of Adds.	No. of Muls.
Example-1					
MLANN	31	0	261	230	230
FLANN	1	14	15	14	15
CFLANN	2	10	12	11	12
Example-2					
MLANN	31	0	281	250	250
FLANN	1	24	25	24	25
CFLANN	2	17	19	18	19
Example-3					
MLANN	62	0	522	460	460
FLANN	2	38	40	39	40
CFLANN	2	14	16	15	16
Example-4					
MLANN	31	0	341	310	310
FLANN	1	20	21	20	21
CFLANN	2	13	15	14	15

3.5 Conclusion

The Chapter has introduced a new Cascaded FLANN(CFLANN) model with its learning algorithm. This adaptive structure is then used to identify nonlinear complex plants. Computer simulation based experiments have been carried out to validate its performance and compare the same with those obtained by other standard methods. The results of simulation indicate that the proposed CFLANN structure involves least computation and offers best performance compared to those offered by the existing FLANN and MLANN based methods.

References

[3.1] Chia-Feng Juang, "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms", IEEE Trans. on Fuzzy Systems, vol. 10, no. 2, pp. 155-170, April 2002.

[3.2] A. Lo Schiavo and A. M. Luciano, "Powerful and flexible fuzzy algorithm for nonlinear dynamic system identification", IEEE Trans. on Fuzzy Systems, vol. 9, no. 6, pp. 828-835, December 2001.

[3.3] N. V. Bhat, P. A. Minderman Jr., T McAvoy and N. S. Wang., "Modeling chemical process systems via neural computation, IEEE Contr. Syst. Mag., vol. 10, no. 3, pp. 24-29, April 1990.

[3.4] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", IEEE Trans. on Neural Networks, vol. 1, pp. 4-26, January 1990.

[3.5] D. N. Nguyen and B. Widrow, "Neural networks for self learning control system", Int. J. Contr., vol. 54, no. 6, pp. 1439-1451, 1991.

[3.6] G. Cembrano, G. Wells, J. Sarda and A. Ruggeri, "Dynamic control of a robot arm based on neural networks", Contr. Eng. Practice, vol. 5, no. 4, pp. 485-492, 1997.

[3.7] S. Lu and T. Basar, "Robust nonlinear system identification using neural network models", IEEE Trans. on Neural Networks, vol. 9, pp. 407-429, May 1998.

[3.8] S. Chen, S. A. Billings and P. M. Grant, "Recursive hybrid algorithm for nonlinear system identification using radial basis function networks", Int. J. Contr., vol. 55, no. 5, pp. 1051-1070, 1992.

[3.9] S. V. T. Elanayar and Y. C. Shin, "Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems", IEEE Trans. on Neural Networks, vol. 5, pp. 594-603, July 1994.

[3.10] E. J. Hartman, J. D. Keeler and J. M. Kowalski, "Layered neural networks with Gaussian hidden units as universal approximation", Neural Comput., vol. 2, pp. 210-215, 1990.

[3.11] J. Zhang, G. G. Walter, Y. Miao and W. G. W. Lee, "Wavelet neural networks for function learning", IEEE Trans. Signal Processing, vol. 43, pp. 1485-1497, June 1995.

[3.12] Qinghua Zhang, "Using wavelet network in nonparametric estimation", IEEE Trans. on Neural Network, vol. 2, no. 2, pp. 227-236, 1997.

[3.13] M. K. Tsatsanis and G. B. Giannakis, "Time varying system identification and model validation using wavelets", IEEE Trans. on Signal Processing, vol. 41, no. 12, pp. 3512-3523, 1993.

[3.14] Yonghong Tan, Xuanju Dang, Feng Liang and Chun-Yi Su, "Dynamic wavelet neural network for nonlinear dynamic system identification", Proc. of the 2000 IEEE International Conf. on Control Applications, Anchorage, Alaska, USA, September 25-27, 2000, pp. 214-219.

[3.15] Y. Chen and S. Kawaji, "Evolving wavelet neural networks for system identification ", Proc. of International Conf. on Electrical Engineering, Kitakyushu, Japan, 2000, pp. 279-282.

[3.16] Y. Chen and S. Kawaji, "Evolving the basis function neural networks for system identification", Int. J. Adv. Comput. Intell, vol. 5, no. 4, pp. 229-238, 2001.

[3.17] P. S. Sastry, G. Santharam and K. P. Unnikrishnan, "Memory neural networks for identification and control of dynamical systems", IEEE Trans. on Neural Networks, vol. 5, pp. 306-319, March 1994.

[3.18] A. G. Parlos, K. T. Chong and A. F. Atiya, "Application of recurrent multilayer perceptron in modeling of complex process dynamics", IEEE Trans. on Neural Networks, vol. 5, pp. 255-266, March 1994.

[3.19] P. A. Mastorocostas and John B. Theocharis, "A recurrent fuzzy-neural model for dynamic system identification", IEEE Trans. on Systems, Man and Cybernetics-Part B: Cybernetics, vol. 32, no. 2, pp. 176-190, April 2002.

[3.20] Chen-Sen Ouyang and Shie-Jue Lee, "An improved TSK-type recurrent fuzzy network for dynamic system identification", Proc. of IEEE International Conf. on Systems, Man and Cybernetics, Washington, USA, Oct. 5-8, 2003, vol. 4, pp. 3342-3347.

[3.21] Yen-Ping Chen and Jeen-Shing Wang, "A novel recurrent neural network with minimal representation for dynamic system identification", Proc. of IEEE International Joint Conf. on Neural Networks, July 25-29, 2004, vol. 2, pp. 849-854.

[3.22] Jeen-Shing Wang and Yen-Ping Chen, "A fully automated neural network for unknown dynamic system identification and control", IEEE Trans. on Circuits and Systems-I, vol. 53, no. 6, pp. 1363-1372, June 2006.

[3.23] Y. H. Pao, *Adaptive Pattern Recognition and Neural Network*. Reading, MA:Addision-Wesley, 1989.

[3.24] A. Namatame and N. Ueda, "Pattern classification with Chebyshev neural network", Int. J. Neural Network, vol. 3, no. 4, pp. 23-31, March 1992.

[3.25] B. Igelnik and Y. H. Pao, "Stochastic choice of basis functions in adaptive function approximation and the functional link net", IEEE Trans. on Neural Network, vol. 6, no. 6, pp. 1320-1329, Nov. 1995.

[3.26] T. T. Lee and J. T. Jeng, "The Chebyshev polynomial based unified model neural networks for function approximations", IEEE Trans. on System, man and Cybernetics-B, vol. 28, pp. 925-935, December 1998.

[3.27] Y. H. Pao, S.M. Phillips and D. H. Sobajic, "Neural net computing and intelligent control systems", Int. J. Contr., vol. 56, no. 2, pp. 263-289, 1992.

[3.28] J. C. Patra, R. N. Pal, R. Baliarsingh and G. Panda, "Nonlinear channel equalization for QAM signal constellation using artificial neural networks", IEEE Trans. on Systems, Man and Cybernetics- Part B, vol. 29, no.2, pp. 262-271, April 1999.

[3.29] J. C. Patra and R. N. Pal, "A functional link artificial neural network for adaptive channel equalization", Signal processing, vol. 43, pp. 181-195, May 1995.

[3.30] J. C. Patra, R. N. Pal, B. N. Chatterjee and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks", IEEE Trans. on Systems, Man and Cybernetics – Part B, vol. 29, no. 2, pp. 254-262, April 1999.

[3.31] J. C. Patra and A. C. Kot, "Nonlinear dynamic system identification using Chebyshev functional link artificial neural networks", IEEE Trans. on Systems, Man and Cybernetics – Part B, vol. 32, no. 4, pp. 505-511, August 2002.

[3.32] S. Purwar, I. N. Kar and A. N. Jha, "Online system identification of complex systems using Chebyshev neural networks", Applied Soft Computing, 7, pp. 364-372, 2007.

[3.33] J. Teeter and M. Y. Chow, "Application of functional link neural network to HVAC thermal dynamic system identification", IEEE Trans. Ind. Electron., vol. 45, no. 1, pp. 170-176, Feb. 1998.

[3.34] B. S. Lin, F. C. Chong and F. Lai, "A functional link network with higher order statistics for signal enhancement", IEEE Trans. on Signal Processing, vol. 54, no. 12, pp. 4821-4826, December 2006.

[3.35] J. C. Patra, G. Panda and R. Baliarsingh, "Artificial Neural Network based nonlinearity estimation of pressure sensor", IEEE Trans. on Inst. And Measurement, vol. 43, no. 6, pp.874-881, Dec., 1994.

[3.36] S. Mishra and G. Panda, "A novel method for designing LVDT and its comparison with conventional design", IEEE Sensor application Symposium, Texas, USA, 7-9 Fe., 2006, pp. 129-134.

[3.37] R. Majhi, G. Panda and G. Sahoo, "Efficient prediction of foreign exchange using single layer artificial neural network", Proc. of IEEE Conf. on Cybernetics and Intelligent Systems, Bankok, June 2006, pp. 1-5.

[3.38] M. Pachter and O. R. Reynolds, "Identification of a discrete time dynamical system", IEEE Trans. Aerospace Electronic System, vol. 36, issue 1, pp. 212-225, 2000.

Chapter

4

Identification of IIR Plants using Comprehensive Learning Particle Swarm Optimization

4.1 Introduction

DURING the last two decades adaptive infinite-impulse response (IIR) filtering and identification have been an active field of research. They have been applied to linear prediction, adaptive differential pulse code modulation, channel equalization, process control, echo cancellation, adaptive array processing and intelligent instrumentation. Many real world systems such as speech synthesis and recognition, acoustical modeling and adaptive digital subscriber loop (ADSL) are recursive in nature and it is advantageous to model such plants using IIR adaptive filters. The main advantage of IIR system is that it provides significantly improved performance than an adaptive FIR filter having the same number of coefficients. This is due to the fact that the output feed back generates an infinite impulse response with a finite number of parameters. Further the

output of an IIR filter approximates the desired response more effectively if both poles and zeros are present in a filter. Alternatively, to achieve a pre-specified performance, an IIR filter requires considerably fewer coefficients than the corresponding FIR filter. It is a fact that the adaptive IIR filters effectively substitute the conventionally-used adaptive FIR filters for many practical applications [4.1].

As the error surface of IIR filters is usually multimodal with respect to the filter coefficients, the gradient based learning algorithms such as the least-mean-square (LMS) very often get stuck at local minima and its associated weights do not converge to the global optimum [4.2]. This algorithm tries to find out the minimum point of the error surface by moving in the direction of negative gradient. Like most of the learning algorithms it may lead the filter to a local minimum when the error surface is multimodal. In addition, the convergence behavior depends on the choice of the step size and initial values of filter coefficients.

The adaptive IIR filtering has two distinct approaches which correspond to different formulations of prediction error. These are equation-error [4.3, 4.4] and output-error formulations [4.5, 4.6]. In the equation-error method the feed back coefficients are updated in an FIR form which are then copied to a second filter implemented in all-pole form. This formulation is essentially a type of adaptive FIR filtering. However this approach may lead to biased estimates of the filter coefficients. On the other hand the output-error formulation updates the coefficients of the feedback path directly in a pole-zero recursive form. This approach does not generate biased estimates of the coefficients. But the adaptive algorithm may converge to a local minimum of the mean square error (MSE) leading to an incorrect estimate of the coefficients. In addition, its convergence properties are not easily predicted [4.10]. Out of the two formulations of adaptive IIR filtering, the output error based approach provides improved performance in system identification if the problem of local minima associated with this algorithm is overcome.

Thus the main motive of this chapter is to propose a new adaptive algorithm using a population based bio-inspired technique known as particle swarm optimization(PSO) for identification of IIR or pole-zero systems which is expected to overcome the local minima problem and to provide accurate estimates of the pole-zero coefficients. To improve the performance of complex multimodal problems, further a comprehensive learning PSO (CLPSO) algorithm has recently been proposed [4.9]. In this chapter the identification of IIR systems is also carried out using this structured stochastic search algorithm. Since the

proposed technique is independent of the adaptive filter structure and is capable of converging to the global solution for multimodal problems, it is expected to be a potential candidate for identification of IIR systems.

4.2 Related work

Swarm Intelligence (SI) is an artificial intelligence technique which involves the study of collective behavior in decentralized and self organized systems. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. Although there is no centralized control structure dictating how individual agent should behave, local interactions between such agents lead to the emergence of global behavior. Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, honey bees and fish schooling. The SI refers to the problem solving behavior which emerges from the interaction between the individuals of such system. The algorithmic models of such behaviors have shown to adapt well in changing environments and are flexible and robust. The last decade has witnessed rapid growth of research interests in various SI paradigms, one of which is particle swarm optimization (PSO) [4.7, 4.8]. The PSO is a global optimization algorithm for dealing with problems in which the best solution can be represented as a point or surface in an n-dimensional space. Its main advantage over other optimization strategies such as simulated annealing [4.17] is that the large number of members that make up the particle swarm enable the technique to be resilient to the problem of local minima. The PSO algorithm is easy to implement and has been shown to perform well for many optimization problems.

In [4.6], Johnson presented a tutorial on adaptive IIR filtering techniques highlighting the common basis between filtering and system identification. Subsequently another tutorial on adaptive IIR filtering was published [4.11-14] that dealt with different algorithms, error formulations and realizations. These filters are mostly direct-form realization. The direct form is a convenient and simple structure but can not ensure stability of the adaptive filter. To overcome this problem, the parallel [4.15] and lattice [4.16] forms have been proposed. These structures offer simple stability monitoring with less complexity than that of direct form. A review paper has been reported [4.26] on adaptive IIR filtering algorithms for system identification using a unifying frame work. For achieving efficient system identification neural network has also been introduced in the literature [4.55-4.56]

The drawback of all adaptive IIR filter structures is that they produce error surfaces that inherently tend to be multimodal. Therefore the local optimization techniques such as the gradient descent algorithms, are not suitable because they are likely to be trapped to local minimum solution. An alternative to gradient based techniques is a structured stochastic search of the error surface. These type of global searches are structure independent because a gradient is not calculated and the filter structure does not directly influence the parameter updates. Due to this feature, these types of algorithms are potentially capable of globally optimizing IIR filter structures. Several structured stochastic search approaches have appeared in the literature, using simulated annealing [4.17], genetic algorithm (GA) [4.18] and PSO [4.7, 4.8]. An adaptive genetic algorithms for determining the optimum filter coefficients in a recursive adaptive filter is presented in [4.19]. The GA has also been applied to optimize the parameters of adaptive IIR filters [4.20]. In another publication [4.21] GA based approach applied for system identification and control of both continuous and discrete time systems has been reported. For efficient adaptive IIR filtering a fast genetic search algorithm has been introduced [4.22] in the LMS algorithm. In another paper [4.23] the nonlinear parameters of the IIR filters have been estimated using GA. A hierarchical GA based algorithm is proposed in [4.24] for design and optimization of IIR filter structure. In a recent article [4.25] a new learning algorithm is introduced embedding the genetic search into the gradient descent algorithm to accelerate the learning process and to provide global search capability. It is reported that this method outperforms the LMS algorithm and the gradient lattice algorithm in terms of convergence speed and ability to locate the global optimal solution.

The basics of PSO is dealt in Section 2.5.2. The research papers on PSO which have appeared in the literature have focused on two fronts : improving the performance of PSO [4.27-4.37] by incorporating a number of modifications in the algorithm and application of these PSO algorithms in diverse fields such as minimization of functions of many variables [4.38], image segmentation [4.39, 4.40], design of antennas [4.41] and stock market prediction [4.42], design of tree structures [4.43], learning to pay games [4.44] and multimodal biomedical image reistration [4.45] and design of adaptive IIR filter [4.46]. Various variants of PSOs are PSO with decreasing inertia weights [4.27], PSO with fuzzy adaptive inertia weights [4.28], self-organizing hierarchical PSO with time varying accelerating coefficients [4.29], PSO with linearly decreasing $V_{\max}$ [4.30], PSO with

constriction factor [4.31], local version of PSO with constriction factor [4.32], dynamic neighborhood PSO [4.33], unified PSO [4.34], fully informed PSO [4.35], fitness-distance-ratio based PSO [4.36] and cooperative PSO [4.37]. Some researchers have proposed hybridization by combining PSO with other search techniques to improve the performance of the PSO. Evolutionary operators such as selection, crossover and mutation have also been suggested in PSO to retain the best particles [4.47] and to improve the ability to escape from local minima [4.48]. Two recent reported works in this direction are PSO with crossover [4.49] and PSO with mutation [4.50]. In order to maintain the diversity and to escape from local optima, relocation of particles when these are too close to each other [4.51] or use of some collision-avoiding mechanisms [4.52] have been proposed in the literature. Negative entropy has been employed in PSO [4.53] to discourage premature convergence. Deflection, stretching and repulsion mechanisms have been introduced in [4.54] to find as many minima as possible by preventing particles from entering to previously discovered minimal region.

4.3 Basics of modified PSO and CLPSO algorithms

The introduction of constriction factor K in the velocity equation (2.48) of conventional PSO has improved the convergence of the PSO algorithm [4.31]. Accordingly the velocity equation is modified as

$$V_i(d) = K * V_i(d) + c_1 * rand1_i(d) * (P_i(d) - X_i(d)) + c_2 * rand2_i(d) * (P_g(d) - X_i(d)) \quad (4.1)$$

where $K = \dfrac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}$ (4.2)

and $\phi = c_1 + c_2,\ \phi > 4$ (4.3)

Usually ϕ is set to 4.1 and $K = 0.729$. As a result each of the $(P_i - X_i)$ terms is calculated by multiplying 0.729 * 2.05= 1.49445.

In the original PSO, each particle learns from its *pbest* and *gbest* simultaneously. In the PSO, the social learning aspect is restricted only to the *gbest*. This appears to be somewhat an arbitrary decision. In addition, all particles in the swarm learn from the *gbest* even if the current *gbest* is far from the global optimum. Under such situations, the particles are attracted easily and trapped in to an inferior local optimum. As the fitness

value of the particle is decided by all dimensions, a particle which has discovered the value corresponding to the global optimum in one dimension may have a low fitness value because of poor solution in other dimensions. In order to prevent this Liang et al have proposed [4.9] a novel learning strategies which differ mainly in three aspects compared to the PSO reported in [4.7], [4.8].

(i) Instead of using particle's own $pbest$ and $pbest$ as the exemplars, all particles' $pbest$ are used as exemplars to guide a particle's flying direction

(ii) Each dimension of a particle may learn from the corresponding dimension of different particles' $pbest$.

(iii) Instead of learning from two exemplars ($gbest$ and $pbest$) at the same time in very generation as in the original PSO, each dimension of the particle trained from just one exemplar for a few generations.

In the CLPSO algorithm the velocity update equation [4.9] is given by

$$V_i(d) = w * V_i(d) + c * rand_i(d)(pbest_{f_i(d)}(d) - X_i(d)) \tag{4.4}$$

where $f_i(d)$ gives which particles' $pbest$s will be used for the ith particle. The term $pbest_{f_i(d)}(d)$ represents the corresponding dimension of a particle's $pbest$ including its own and such a decision depends on the learning probability P_c. For each dimension of particle i, a random number is generated. If this number is larger then P_c, the corresponding dimension learns from its own $pbest$, otherwise it learns from another particle's $pbest$ until the particle ceases improving for a certain number of generations called the refreshing gap m, then f_i is reassigned for the particle. In the later case the tournament selection procedure is employed. The above stated operations increase the diversity of swarms which results in enhanced performance when solving complex multimodal problems. It is reported that different values of P_c yielded a solution which is different from that obtained by taking same P_c for all particles. If different values of P_c for different particles are taken then the particles have different levels of exploration and exploitation ability in the population. The empirical relation for ith particle is given by

$$P_{c_i} = 0.05 + 0.45 * \frac{(\exp(\frac{10(i-1)}{P_s - 1}) - 1)}{(\exp(10) - 1)} \tag{4.5}$$

where P_s = size of population. In many practical problems bounds are imposed on the ranges of the variables. Two search ranges are suggested :

(i) $[X_{\min}, X_{\max}]$ and

(ii) $\min(X_{\max}(d), \max(X_{\min}(d), X_i(d))$

The refreshing gap parameter m also influences the results as it affects the convergence velocity. The details of the CLPSO algorithms is available in [4.9].

4.4 Adaptive system identification of IIR systems

The block diagram of an adaptive system identification of an IIR plant is shown in Fig. 4.1. The model is an output-error adaptive IIR filter and is characterized by the recursive difference equation given in (4.6)

$$\hat{y}(n) = \sum_{m=1}^{N-1} \hat{a}_m(n)\hat{y}(n-m) + \sum_{m=0}^{M-1} \hat{b}_m x(n-m) \tag{4.6}$$

where $x(n)$ and $\hat{y}(n)$ represent the n th input and output of the plant respectively. The present estimated output $\hat{y}(n)$ depends on the past estimated output samples $\hat{y}(n-m), m = 1, 2, ...N-1$. $\{\hat{a}_m(n), \hat{b}_m(n)\}$ represent adjustable coefficients which at the end of the adaptation process give the estimated pole-zero parameters of the IIR plant.

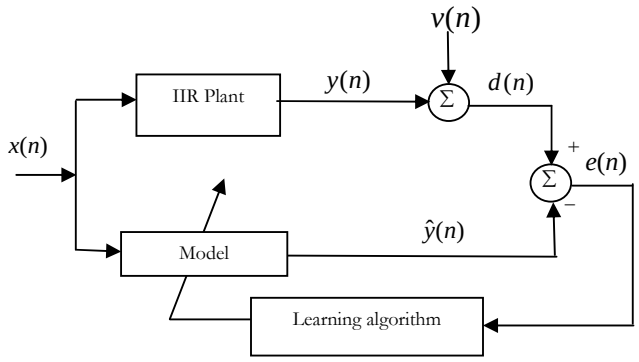


Fig. 4.1 Adaptive identification of IIR systems using output-error adaptive IIR filter as the model

The output $d(n)$ of the plant is represented by (4.7)

$$d(n) = \sum_{m=1}^{N-1} a_m(n) y(n-m) + \sum_{m=0}^{M-1} b_m(n) x(n-m) + v(n) \tag{4.7}$$

where $y(n)$ and $v(n)$ denote the output and measurement noise respectively. This noise is uncorrelated with the input $x(n)$. The pole and zero parameters of the IIR plant are $a_m(n)$ and $b_m(n)$ respectively. In the system identification configuration of Fig. 4.1, the model is represented by output-error adaptive IIR filter of the form

$$\hat{y}(n) = \left(\frac{\hat{B}(n,z)}{1-\hat{A}(n,z)} \right) x(n) \tag{4.8}$$

where the feed forward and feed back transfer functions are given by

$$A(n,z) = \sum_{m=1}^{N-1} \hat{a}_m(n) z^{-m} \text{ and } \hat{B}(n,z) = \sum_{m=0}^{M} \hat{b}_m(m) z^{-m} \tag{4.9}$$

respectively.

Unlike the equation-error formulation, the pole-polynomial $1-\hat{A}(n,z)$ is adapted directly in an IIR filter form. Such formulation is a natural generalization of the adaptive FIR filter where $\hat{A}(n,z) = 0$. Equation (4.6) may also be written as an inner product form

$$\hat{y}(n) = \underline{\hat{\theta}}^T(n) \underline{\phi}(n) \tag{4.10}$$

where the estimated coefficient vector $\underline{\hat{\theta}}(n)$ and the signal vector $\underline{\phi}(n)$ are given by (4.11)

$$\underline{\hat{\theta}}(n) = [\hat{a}_1(n), \ldots\ldots\ldots\ldots, \hat{a}_{N-1}(n), \hat{b}_0(n), \ldots\ldots\ldots, \hat{b}_{M-1}(n)]^T$$

$$\text{and } \underline{\phi}(n) = [\hat{y}(n-1), \ldots \hat{y}(n-N+1), x(n), \ldots x(n-M+1)]^T \tag{4.11}$$

The output $\hat{y}(n)$ is a nonlinear function of $\underline{\hat{\theta}}(n)$ because the delayed output signals $\hat{y}(n-k)$ of $\underline{\phi}(n)$ depend on previous coefficient values. The output error is given by $e(n) = d(n) - \hat{y}(n)$ and is generated by subtracting the estimated output in (4.6) from $d(n)$. It is evident that $e(n)$ is a nonlinear function of θ and hence the mean square output error is not a quadratic function and therefore it can have multiple minima. The gradient based adaptive algorithms like the LMS could converge to one of the local solutions yielding inaccurate estimates of pole zero parameters. The CLPSO algorithm is therefore has been employed in this chapter in updating those parameters of the model so that the parameter estimates would be optimal.

4.5 CLPSO based identification of IIR systems

The identification of IIR system is formulated as an optimization problem. The steps involved in the CLPSO based algorithm for identification are the following :

Step-1 Input samples of a suitable window length (L) are selected from a zero mean uniform random sample. It is represented by $x(l)$, where $l = 0,.....,L-1$ and lies between +0.5 to -0.5.

Step-2 $D = (M + N - 1)$ number of random numbers are generated which represents the initial position of particle X . The first $(N-1)$ random numbers denote the initial feedback parameters and the remaining M random numbers represent the feed-forward parameters of the model of the plant. Another set of $M + N - 1$ random numbers are also generated to represent the corresponding velocities V of the particles. Here D denotes dimension of the particle.

Step-3 The procedure in Step-2 is repeated for a specified population size P_s. The complete set of population constitutes a swarm.

Step-4 Input samples of window size (L) are applied sample by sample to the plant of known coefficients and then added with noise to generate the desired signal $d(l)$.

Step-5 The same input is also applied to the model sequentially to get the output signal $\hat{y}_i(l)$ which is the estimated output of the model corresponding to the l th input sample and for the i th particle.

Step-6 The Mean Square Error (MSE) of i th particle (which represents the cost function) is computed using $J_i = \frac{1}{L}[\sum_{i=0}^{L-1}(d(l) - \hat{y}_i(l))^2$ (4.12)

Step-7 In the same way the cost functions of all other particles are also evaluated for every generation. The particle giving the minimum cost function, provides the best possible representation of the unknown plant to be modeled.

Step-8 The *pbest* represents the best positions (i. e. set of model parameters that gives the minimum cost function value) for a particular particle. It is initialized as the initial position of the particle. The *gbest* represents the best position in the swarm. It is initialized as the position of the particle which gives the minimum cost function value in the swarm.

Step-9 k is the iteration number initialized to 1. The first particle is chosen i. e. $i = 1,$ where i is the particle number. Linearly decreasing inertia weight within the range 0.9 to 0.4 is used for updating the velocity and position of particles in each iteration. The inertia weight at k th generation is given by

$$w_k = w_0 - \frac{(w_0 - w_1) * k}{itr} \tag{4.13}$$

where k =generation counter (from 1 to itr)

itr =number of iterations

$w_0 = 0.9$ and $w_1 = 0.4$

Step-10 The refreshing gap parameter m is adjusted depending on the function or problem to be optimized. The $flag_i$ represents the number of generations the i th particle has not improved its own $pbest$. The flag is initialized to 0 for all particles.

Step-11 Another parameter Pc_i called as the learning probability of a particle is initialized according to the empirical formula given in (4).

Step-12 If $flag_i < m$ then go to Step-(20).

Step-13 Starting with $d = 1$ for every dimension of a particle a random number is generated (rand).

Step-14 If $rand < Pc_i$ then go to Step-17 else to Step-15.

Step-15 Set $f_i(d) = i$, which gives the number of the particle whose $pbest$ will be used for the present particle.

Step-16 Now if $d < D$ then increment d and go to Step-13 or else go to Step-19.

Step-17 Two particles are selected randomly by using

$$\begin{aligned} f1_i(d) &= \lceil rand1_i(d) * P_s \rceil \\ f2_i(d) &= \lceil rand2_i(d) * P_s \rceil \end{aligned} \tag{4.14}$$

where P_s = population size

$\lceil \ \rceil$ = ceiling operator

Step-18 Subsequently the cost function for the two randomly selected particles $f1_i(d)$ and $f2_i(d)$ are computed using their respective $pbest$.

If $J_{(f1_i(d)\,pbest)} < J_{(f2_i(d)\,pbest)}$, then $f_i(d) = f1_i(d)$ else $f_i(d) = f2_i(d)$. Then go to Step-16.

Step-19 Reinitialize $flag_i$ to zero.

Step-20 d is set to 1 representing the dimension of the i th particle.

Step-21 New velocities and positions are calculated using

$$\begin{aligned} V_i(d) &= w_k * V_i(d) + c * rand_i(d) * (pbest_{f_i(d)}(d) - X_i(d)) \\ V_i(d) &= \min(V_{\max}(d), \max(V_{\min}(d), V_i(d)) \\ X_i(d) &= X_i(d) + V_i(d) \end{aligned} \tag{4.15}$$

where c = acceleration coefficients

Step-22 If $d < D$ then increment d and go to Step-21 else go to Step-23.

Step-23 If $X_i(d) \in [X_{\min}, X_{\max}]$ then go to Step-24 else go to Step-27.

Step-24 Calculate the cost function for new position of particle.

Step-25 If $J_{X(i)} < J_{pbest(i)}$ for the i th particle then go to Step-26 else increment flag and go to Step-27.

Step-26 Reinitialize $pbest(i) = X(i)$ and also initialize $flag(i) = 0$. If $J_{X(i)} < J_{gbest}$ then $gbest = X(i)$.

Step-27 If $i < P_s$ then increment i and go to Step-12 or else go to Step-28.

Step-28 If $k < itr$, then increment k and go to Step-9.

Step-29 The elements corresponding to $gbest$ particle provide the estimated coefficients of the IIR system.

4.6 Simulation study

By conducting simulation experiments on identification of four bench mark IIR systems (2^{nd} order to 5^{th} order) the performance of proposed CLPSO based method is compared with those obtained from three other standard methods, recursive LMS (RLMS), GA and PSO. The new algorithm is used in adaptive IIR identification to improve the performance of the existing algorithms especially when the error surface is multimodal. The block diagram of Fig. 4.1 is simulated using output-error formulation. The plant (the unknown system) is a fixed IIR filter with transfer function $H(z)$, while the adaptive system is an adaptive IIR filter with transfer function $\hat{H}(z)$ whose coefficients are updated by different learning algorithms. The transfer function of the plant is represented by

$$H(z) = \frac{\sum_{j=0}^{M} b_j z^{-i}}{1 - \sum_{i=1}^{L} a_i z^{-i}} \tag{4.16}$$

In the present simulation both full-order and reduced-order modeling are considered. Local minima phenomena are observed in the reduced-order modeling, while the full-order modeling is used to demonstrate the fast convergent behavior and global search ability of the proposed algorithm. The CLPSO based algorithm discussed in the previous section is used to compute the best estimate of the pole-zero parameters. The input is a zero mean white random signal with uniform distribution. The additive noise $v(n)$ is a white random process uncorrelated with $x(n)$ and with 30dB SNR. The Initial common parameters used for CLPSO, PSO, GA and LMS are listed below :

CLPSO : D = no. of weights to be optimized, P_s = Population size =40 to 150, $X_{\min}$ = lower bound of weights = -1.3, $X_{\max}$ = upper bound of weights = 1.3, $V_{\max}$ = maximum velocity = $X_{\max}$, w_0 = max inertia weight = 0.9 , w_1 = minimum inertia weight = 0.4, c = acceleration factor = 1.042, X = positions of particles and V = velocities of the particles, m = refreshing gap = 5 and L = 25.

PSO : D = no. of weights to be optimized, P_s = Population size=120 to 400 X = positions of particles and V = velocities of the particles, K =constriction factor = 0.729, $c_1 = c_2$ = acceleration coefficients = 1.49445 and L = 500.

GA : D = no. of weights to be optimized, P_s = Population size = 80 to 120, N = no. of bits = 20 to 40, $W_{\max}$ = max. of weights , $W_{\min}$ = min of weights, P_c = probability of crossover =0.9, P_m = probability of mutation = 0.01, L = 1000

LMS : D = no. of weights to be optimized, μ = convergence coefficients = 0.1, L = 10, 000.

Example-1 The plant is a second order IIR filter [4.46] with M=1, L=2. The difference equations of plant : $y(n) = \sum_{i=1}^{2} a_i y(n-i) + \sum_{j=0}^{1} b_j x(n-j)$

$d(n) = y(n) + v(n)$

$\{a_i\} = \{0.3, -0.4\}, \{b_j\} = \{1.25, -0.25\}$

Full-order model : $\hat{y}(n) = \sum_{i=1}^{2} \hat{a}_i y(n-i) + \sum_{j=0}^{1} \hat{b}_j x(n-j)$

Reduced – order model : $\hat{y}(n) = \sum_{i=1}^{1} \hat{a}_i y(n-i) + \sum_{j=0}^{0} b_j x(n-j)$

The convergence characteristics of using GA, PSO and CLPSO based training are shown in Fig. 4.2(a) for full-order and in Fig. 4.2(b) for reduced order. Fig. 4.2(a) reveals that GA and PSO can not reach the minimum MSE even after 500 generations where as the new algorithm can converge to the optimal solution with a MSE level of 10^{-5} in only 200 generations. Similarly the convergence plot of Fig. 4.2(b) for reduced order model exhibits superior MSE performance of CLPSO compared to other two. The results clearly show that for reduced order the multimodal situation does not affect the convergence performance of CLPSO but does so in other two cases.

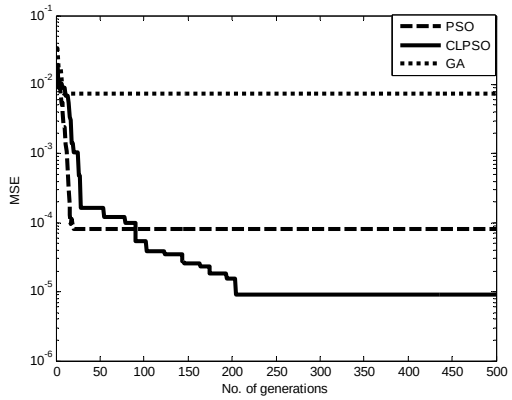


Fig. 4.2(a) Comparison of convergence characteristics of different methods for an exact 2nd order IIR model

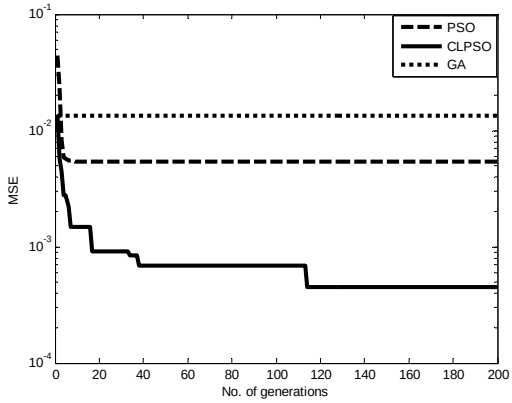


Fig. 4.2(b) Comparison of convergence characteristics of different methods for a reduced order (1st order) IIR model

Example-2 The adaptive system is a third-order IIR filter [4.25] to model a plant of the same order $(M = 2, L = 3)$ with $\{a_i\} = \{0.6, -0.25, 0.2\}, \{b_j\} = \{-0.2, -0.4, 0.5\}$. The convergence plots of full-order and reduced-order $(M = 1, L = 2)$ are shown in Fig. 4.3(a) and (b) respectively. The plots indicate that the convergence performance of the new method is superior to the GA and PSO based methods under multimodal situation (Fig. 4.3(b)) and as well as when full-order models are used. The CLPSO method clearly exhibits best performance.

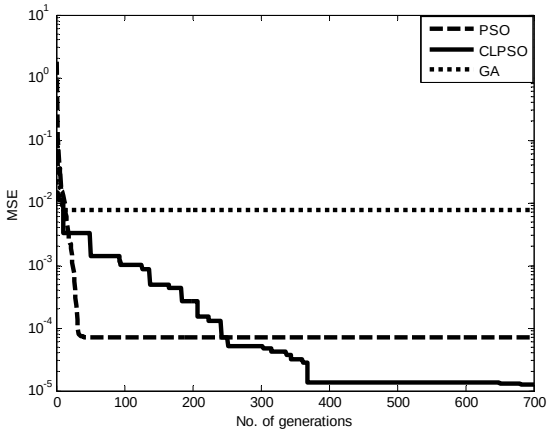


Fig. 4.3(a) Comparison of convergence characteristics of different methods for an exact 3rd order IIR model

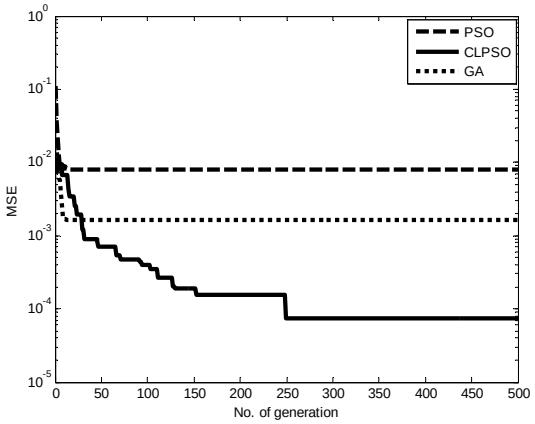


Fig. 4.3(b) Comparison of convergence characteristics of different methods for a reduced order (2nd order) IIR model

Example-3 In this experiment the plant is of fourth order IIR system [4.15] $\{a_i\} = \{-0.04, -0.2775, 0.2101, -0.14\}, \{b_j\} = \{1, -0.9, 0.81, -0.729\}$ The adaptive IIR system is having $M = 3, L = 4$ for full-order model and $M = 2, L = 3$ for reduced order model. The convergence characteristics for the two models are shown in Fig. 4.4 (a) and (b). It is observed that standard GA and PSO exhibits faster convergence initially, but they fail to improve further because the chromosomes and the swarm quickly become stagnant and hence lead to suboptimal solution. However the new algorithm does not stagnate allowing it to reach the minimum noise floor level. The convergence plot of Fig. 4.4(b) for reduced order model also reveals that both the GA and PSO based algorithms get trapped to local solution while the new algorithm clearly exhibits superior performance.

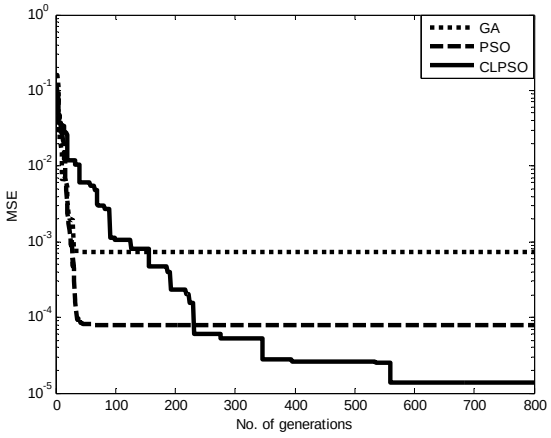


Fig. 4.4(a) Comparison of convergence characteristics of different methods for an exact 4th order IIR model

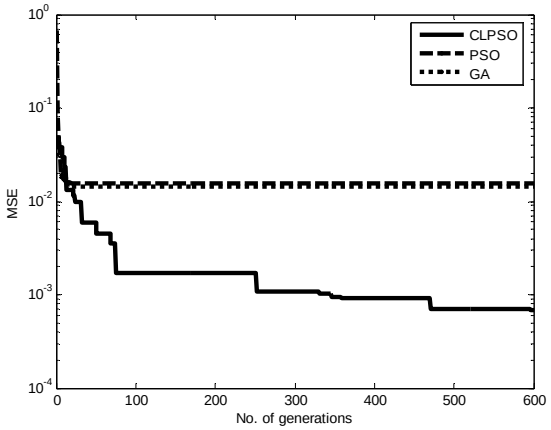


Fig.4.4(b) Comparison of convergence characteristics of different methods for a reduced order (3rd order) IIR model

Example-4 In this example the plant is a fifth order low pass Butterworth IIR filter taken from [4.46]. The plant is represented by

$$y(n) = \sum_{i=1}^{5} a_i y(n-i) + \sum_{j=0}^{5} b_j x(n-j)$$

$$d(n) = y(n) + v(n)$$

$$\{a_i\} = \{-0.9853, -0.9738, -0.3864, -0.1112, -0.0113\}$$
$$\{b_j\} = \{0.1084, 0.5419, 1.0837, 1.0837, 0.5419, 0.1084\}$$

The full-order IIR model is given by $\hat{y}(n) = \sum_{i=1}^{5} \hat{a}_i \hat{y}(n-i) + \sum_{j=0}^{5} \hat{b}_j x(n-j)$

The corresponding reduced order model is also simulated. The convergence behaviors of the exact and reduced-order models are shown in Fig. 4.5(a) and (b) respectively. It is observed that both the full and reduced order GA and PSO based models fail to escape from local minima where as the proposed model is not affected and exhibits significant improvement in the convergence performance in both exact and reduced structures.

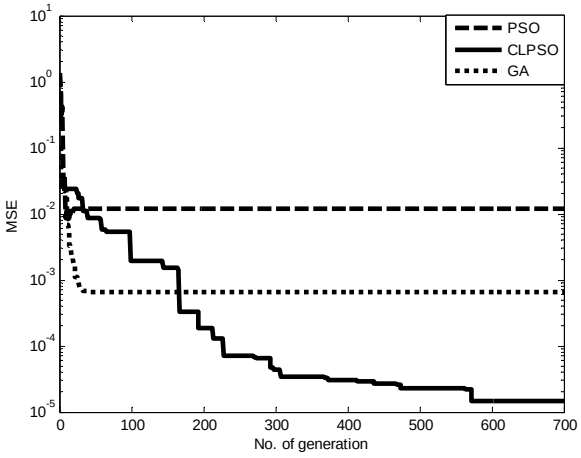


Fig. 4.5(a) Comparison of convergence characteristics of different methods for an exact 5th order IIR model

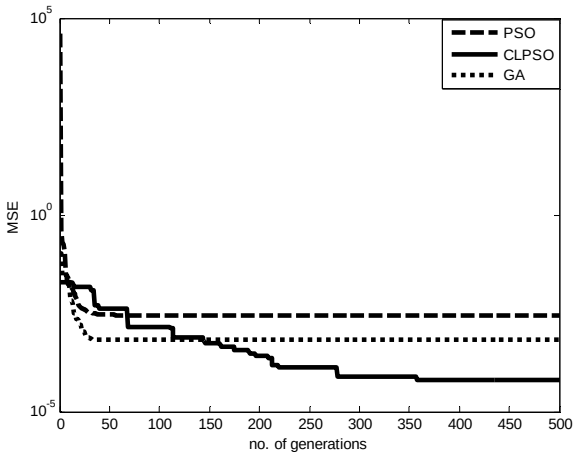


Fig. 4.5(b) Comparison of convergence characteristics of different methods for a reduced order (4th order) IIR model

Table 4.1 summarizes the comparison performance of GA, PSO and CLPSO based methods in terms of minimum MSE after convergence, execution time in seconds, the product of population size and number of input samples used during training. All the three parameters indicate the performance measure of a learning algorithm. In all the examples, these three parameters are observed to be substantially low in case of the new algorithm. For example, GA and PSO based approach take 40 to 80 times and 20 to 150 times more computation of fitness function compared to that required by the new method. The estimated feed forward and feed back coefficients of the IIR systems obtained from RLMS, GA, PSO and CLPSO based methods are also listed in Table 4.2 along with the corresponding plant parameters. In all the cases studied it is observed that the estimated coefficients obtained from the new method are in close agreement with the true coefficients of the plant compared to those obtained by other three methods.

The CLPSO is an improved version of conventional PSO algorithm. The CLSPO is inherently a population based algorithm. As a result its initial rate of convergence is poor. In the IIR identification problem the prime objective is the accuracy of the final solution instead of faster convergence. Being fully aware of such a characteristic the CLPSO was selected for training of the IIR model to provide best identification of IIR plants under multimodal situation. The initial motivation to carry out the research has been successful as may be evident from the set of simulation results provided in various examples.

Table 4.1
Comparison of performance between GA, PSO & CLPSO based training of weights

MSE at -30dB noise			Execution time in second			No. of times $(P_s * L)$ used		
GA	PSO	CL-PSO	GA	PSO	CL-PSO	GA	PSO	CL-PSO
2nd order IIR system								
-21	-41	-50	18.03	2.47	0.27	53	40	1
3rd order IIR system								
-30	-39	-50	40.28	17.42	0.68	80	155	1
4th order IIR system								
-21	-41	-49	146.7	48.24	1.70	48	80	1
5th order IIR system								
-31	-21	-48	62.64	7.06	1.03	40	20	1

Table 4.2
Comparison between true and estimated pole-zero parameters obtained from RLMS, GA, PSO and CLPSO

Actual Parameters	Estimated Parameters at -3odB NSR			
	GA	PSO	CLPSO	RLMS
2nd order IIR system				
1.25	0.9787	1.2513	1.2514	1.2510
-0.25	0.0285	-0.2514	-0.2423	-0.2765
0.3	0.1428	0.2996	0.2959	0.3177
-0.4	-0.4562	-0.4013	-0.4042	-0.4030
3rd order IIR system				
-0.2	-0.1406	-0.1996	-0.1951	-0.2002
-0.4	-0.5012	-0.4110	-0.4051	-0.4005
0.5	0.3222	0.5016	0.5000	0.4903
0.6	0.2352	0.5569	0.5957	0.5897
-0.25	-0.2110	-0.2302	-0.2362	-0.2518
0.2	0.0109	0.1761	0.1966	0.1940
4th order IIR system				
1	0.9801	0.9948	1.0043	1.007
-0.9	-0.7998	-0.8964	-0.8887	-0.8794
0.81	0.7939	0.8093	0.8032	0.7557
-0.729	-0.5838	-0.7290	-0.7277	-0.7301
-0.04	-0.2025	-0.0388	-0.0492	-0.0582
-0.2775	-0.4145	-0.2779	-0.2766	-0.2520
0.2101	0.0755	0.2136	0.2122	0.2579
-0.14	-0.1685	-0.1385	-0.1354	-0.1156

4.7 Conclusion

This chapter has suitably employed the recently developed CLPSO tool to identify the feed-forward and feed-back coefficients of IIR systems. The new identification algorithm is outlined in details and has been applied on few bench mark systems. Simulation study reveals that the proposed method outperforms the existing standard RLMS, GA, PSO based methods in terms of minimum MSE after convergence, execution time and product of population size and number of input samples used in training. Further the new method exhibits significant improvement in convergence behavior when reduced-order models are used compared to those obtained by GA and PSO methods. This clearly indicates that the new method can converge to the optimal solution even under multimodal environment in which local minima problems can be encountered. Therefore the proposed method provides fastest convergence, least training time and best estimates of feed-forward and feed-back coefficients compared to other three methods.

References

[4.1] John J. Shynk, "Adaptive IIR filtering", IEEE ASSP Magazine, April 1989, pp. 4-21.

[4.2] B. Widrow and S. D. Stearns, Adaptive Signal Processing, Englewood Cliffs, NJ : Prentice-Hall, 1985.

[4.3] J. M. Mendel, Discrete Techniques of Parameter Estimation : The Equation Error Formulation, Marcel Dekker, New York, 1973.

[4.4] R. P. Gooch, "Adaptive pole-zero filtering : the equation error approach", Ph. D. diss. Stanford University, 1983.

[4.5] I. D. Landau, Adaptive Control : The model reference approach, Marcel Dekker, New York, 1979.

[4.6] C. R. Johnson, Jr., "Adaptive IIR filtering : current results and open issues", IEEE Trans. on Information Theory, vol. IT-30, no. 2, pp. 237-250, Mar. 1984.

[4.7] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory", in Proc. of 6th Int. symp. Micro machine Human Sci., Nagoya, Japan, 1995, pp. 39-43, 1995.

[4.8] J. Kennedy and R. C. Eberhart, "Particle swarm optimization", in Proc. of IEEE Int. Conf. Neural Networks, 1995, pp. 1942-1948.

[4.9] J. J. Liang, A. K. Qin, Ponnuthurai Nagaratnam Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions", IEEE Trans. on Evolutionary Computation, vol. 10, no. 3, pp. 281-295, June 2006.

[4.10] S. D. Stearns, "Error surfaces of recursive adaptive filters", IEEE Trans. Circuits Systems, vol. CAS-28, no. 6, pp. 603-606, June 1981.

[4.11] P. L. Feintuch, "An adaptive recursive LMS filter", Proc. IEEE, vol. 64, pp. 1622-1624, Nov. 1976.

[4.12] M. G. Larimore, J. R. Treichler and C. R. Johnson, Jr., "SHARF : An algorithm for adapting IIR digital filters", IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-28, pp. 428-440, Aug. 1980.

[4.13] B. Friedlander, "System identification techniques for adaptive signal processing", IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-30, pp. 240-246, April 1982.

[4.14] H. Fan and W. K. Jenkins, "A new adaptive IIR filter", IEEE Trans. Circuits System, vol. CAS-33, pp. 939-947, October 1986.

[4.15] John Shynk, "Adaptive IIR filtering using parallel form realization", IEEE Trans. Acoustic, speech, signal processing, vol. 37, no. 4, pp. 519-533, April 1989.

[4.16] D. Parikh, N. Ahmed and S. D. Stearns, "An adaptive lattice algorithm for recursive filters", IEEE Trans. Acoust., speech, Signal Processing, vol. ASSP-28, pp. 110-111, Feb. 1980.

[4.17] R. Nambiar and P. Mars, "Genetic and Annealing Approaches to adaptive digital filtering", Proc. 26th Asilomar Conf. on Signals, Systems and Computers, vol. 2, Oct. 1992, pp. 871-875.

[4.18] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning Reading, MA: Addison-Wesley, 1989.

[4.19] D. M. Etter et al, "Recursive adaptive filter design using an adaptive genetic algorithm", Proc. of IEEE conf. on ASSP, 1982, pp. 635-638.

[4.20] R. Nambiar, C. K. K. tang and P. Mars, "Genetic and learning automata algorithms for adaptive digital filters", Proc. of IEEE conf. on ASSP, vol. 4,1992, pp. 41-44.

[4.21] Kristinn Kristinsson and Guy A. Dumont, "System identification and control using genetic Algorithms", IEEE Trans. on Systems, Man and Cybernetics, vol. 22, no. 5, pp. 1033-1046, September 1992.

[4.22] S. C. Ng, C. Y. Chung, S. H. Leung and Andrew Luk, "Fast convergent genetic search for adaptive IIR filtering", IEEE Int. Conf. on Acoustics, Speech and Signal Processing, vol. 3, April 1994, pp. 105-108.

[4.23] Leehter Yao and William A. Sethares, "Nonlinear parameter estimation via the Genetic algorithm", IEEE Trans. on Signal Processing, vol. 42, no. 4, pp. 927-935, April 1994.

[4.24] Kit-sang Tang, Kim-fung Man, Sam Kwong and Zhi-feng Liu, "Design and optimization of IIR filter structure using Hierarchical Genetic Algorithms", IEEE Trans. on Industrial Electronics, vol. 45, no. 3, pp. 481-487, June 1998.

[4.25] S. C. Ng, S. H. Leung, C. Y. Chung, A. Luk and W. H. Lau, " The genetic search approach", IEEE Signal Processing Magazine, Nov. 1996, pp. 38-46.

[4.26] Sergio L. Netto, Paulo S. R. Diniz and Panajotis Agathoklis, "Adaptive IIR filtering algorithms for system identification :A general framework", IEEE Trans. on Edu., vol. 38, no. 1, pp. 54-66, Feb. 1995.

[4.27] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer", in Proc. IEEE Congress on Evolutionary Computation, 1998, pp. 69-73.

[4.28] Y. Shi and R. C. Eberhart, "Particle swarm optimization with fuzzy adaptive inertia weight", in Proc. Workshop Particle Swarm Optimization, Indianapolis, IN, 2001, pp. 101-106.

[4.29] A. Ratnaweera, S. Halgamuge and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time varying accelerating coefficients", IEEE Trans. on Evolutionary Computation, vol. 8, pp. 240-255, June 2004.

[4.30] H. Y. Fan and Y. Shi, "Study on Vmax of particle swarm optimization", in Proc. Workshop Particle Swarm Optimization, Indianapolis, IN, 2001.

[4.31] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability and convergence in a multidimensional complex space", IEEE Trans. Evolutionary Computation, vol. 6, no. 1, pp. 58-73, Feb. 2002.

[4.32] J. Kennedy and R. Mendes, "Population structure and particle swarm performance", in Proc. IEEE Congress on Evolutionary Computation, Honolulu, HI, 2002, pp. 1671-1676.

[4.33] X. Hu and R. C. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization", in Proc. Congr. Evol. Comput., Honolulu, HI, 2002, pp. 1677-1681.

[4.34] K. E. Parsopoulos and M. N. Vrahatis, "UPSO-A unified particle swarm optimization scheme", in Lecture series on Computational Sciences, 2004, pp. 868-873.

[4.35] R. Mendes, J. Kennedy and J. Neves, "The fully informed particle swarm : Simpler, may be better", IEEE Trans. Evolutionary Computation, vol. 8, pp. 204-210, June 2004.

[4.36] T. Peram, K. Veeramachaneni and C. K. Mohan, "Fitness distance ratio based particle swarm optimization", in Proc. Swarm Intelligence Symp., 2003, pp. 174-181.

[4.37] F. van den Bergh abd A. P. Engelbrecht, „A cooperative approach to particle swarm optimization", IEEE Trans. on Evolutionary Computation, vol. 8, pp. 225-239, June 2004.

[4.38] Yanping Lu, Shaozi Li and Changle Zhou, "Multipoint organizational evolutionary algorithm for globally minimizing functions of many variables", in Proc. of IEEE 2nd Int. Conf. on Pervasive Computing and applications, July 2007, pp. 84-89.

[4.39] A. Borji, M. Hamidi and A. M. Eftekhari moghadam, "CLPSO based fuzzy color image segmentation", Annual meeting of the North American fuzzy information processing society, June 2007, pp. 508-513.

[4.40] Weihua Liu, Qingmei Sui, Wei Zhang, Nan Lu and Zhengmin Liu, "Image segmentation with 2-D maximum entropy based on comprehensive learning particle swarm optimization", in Proc. of IEEE Int. Conf. on Automation and Logistics, Aug. 2007, pp. 793-797.

[4.41] S. Baskar, A. Alphones, P. N. Suganthan and J. J. Liang, "Design of Yagi-Uda antennas using comprehensive learning particle swarm optimization", IEE proceeding Microw. Antennas Propag., vol. 152, no. 5, pp. 340-346, October 2005.

[4.42] G. Panda, D. Mohanty, Babita Majhi and G. Sahoo, "Identification of Nonlinear Systems using Particle Swarm Optimization Technique", Proc. of IEEE International Congress on Evolutionary Computation(CEC-2007), Singapore, 25-28, September, 2007, pp.3253-3257.

[4.43] J. F. Schutte and A. a. Groenwold, "Sizing design of truss structures using particle swarms", Strruct. Multidisc. Optim., vol. 25, no. 4, pp. 261-269, 2003.

[4.44] L. Messerschmidt and A. P. Engelbrecht, "Learning to play games using a PSO-based competitive learning approach", IEEE Trans. Evolutionary Computation, vol. 8, pp. 280-288, June 2004.

[4.45] M. P. Wachowiak, R. Smolikova, Y. F. Zheng, J. M. Zurada and A. S. Elmaghraby, "An approach to multimodal biomedical image registration utilizing particle swarm optimization", IEEE Trans. on Evolutionary Computation, vol. 8, pp. 289-301, June 2004.

[4.46] D. J. Krusienski and W. K. Jenkins, "Particle swarm optimization for adaptive IIR filter structures", in Proc. IEEE Congress on Evolutionary Computation, June 2004, pp. 965-970.

[4.47] P. J. Angeline, "Using selection to improve particle swarm optimization", in Proc. IEEE Congress Evolutionary Computation, Anchorage, AK, 1998, pp. 84-89.

[4.48] M. Lovbjerg, T. K. Rasmussen and T. Krink, "Hybrid particle swarm optimizer with breeding and subpopulations", in Proc. Genetic Evol. Comput. Conf., 2001, pp. 469-476.

[4.49] Zhi-feng Hao, Zhi Gang Wang and Han Huang, "A particle swarm optimization algorithm with crossover operator", Proc. of the 6th Int. Conf. on Machine learning and cybernetics, Hong Kong, August 207, pp. 1036-1040.

[4.50] Andrew Stacey, Mirjana Jancic and Ian Grundy, "Particle swarm optimization with mutation", in Proc. of IEEE congress on Evolutionary Computation, December 2003, pp. 1425-1430.

[4.51] M. Lovbjerg and T. Krink, "Extending particle swarm optimizers with seld-organized criticality", in Proc. of IEEE Congress on Evolutionary Computation, Honolulu, HI, 2002, pp. 1588-1593.

[4.52] T. M. Blackwell and P. J. Bentley, "Don't push me! Collision avoiding swarms", in Proc. IEEE Congress on Evolutionary Computation, Honolulu, HI, 2002, pp. 1691-1696.

[4.53] X. Xie, W. Zhang and Z. Yang, "A dissipative particle swarm optimization", in Proc. IEEE Congress on Evolutionary Computation, Honolulu, HI, 2002, pp. 1456-1461.

[4.54] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizes through particle swarm optimization", IEEE Trans. on Evolutionary Computation, vol. 8, pp. 211-224, June 2004.

[4.55] B. B. Murthy and G. Panda, "Efficient neural network algorithms for IIR system identification", in Proc. of International conf. on signal processing applications & technology, Boston, USA, 7-10 October 1996, pp. 1343-1347.

[4.56] B. B. Murthy, C. F. N. Cowan and G. Panda, "Efficient scheme of pole-zero system identification based on multilayer neural network", Electronics Letter, vol. 29, issue 1, pp. 73, Jan. 1993.

Chapter

5

Dynamic System Identification using FLANN Structure and PSO and BFO Based Learning Algorithms

5.1 Introduction

NONLINEAR system identification of complex dynamic plants finds potential applications in many areas of engineering such as control, communication, power system and instrumentation. In recent years, modeling of real time processes has gained significant importance in these areas. Many interesting papers have been reported in the literature to identify both static and dynamic nonlinear systems. The Artificial Neural Network (ANN) has been applied for many identification and control tasks [5.1-5.3] but at the expense of large computational complexity. Narendra and Parathasarathy [5.4] have employed the multiplayer perceptron (MLP) networks for effective identification and

control of dynamic systems such as truck-backer-upper problem [5.5] and robot arm control [5.6]. Subsequently the Radial Basis Function (RBF) network has been introduced [5.7] to develop system identification model of nonlinear dynamic systems [5.8-5.9]. One practical difficulty in this model is the selection of an appropriate set of RBF centres for effective learning. Further the wavelets in place of RBF has been suggested in neural network [5.10-5.11] to develop efficient identification models. The Functional Link Artificial Neural Network (FLANN), a computationally efficient single layer ANN, has been reported in the literature as an useful alterative to MLP for many applications. In the literature the trigonometric [5.12] and Chebyshev [5.13-5.14] based FLANN architecture have been proposed for identification of nonlinear dynamic systems [5.13 -5.14].

The swarm intelligence is the property of a system whereby the collective behavior of unsophisticated agents that are interacting locally with their environment create coherent global functional patterns. This type of intelligence is described by five principles such as proximity, quality, diverse response, stability and adaptability. Swarm intelligence provides a useful paradigm for complementing powerful adaptive systems. Both particle swarm optimization (PSO) and bacterial foraging optimization (BFO) algorithms belong to the family of swarm intelligence and share few computational attributes. These are

(i) Individual elements are updated in parallel

(ii) Each new value depends on its previous value as well as contribution from its neighbors

(iii) All updates are performed according to the same rules.

In recent years, evolutionary computational methods belonging to the swarm intelligence category have proven to be promising tools to solve many engineering and financial problems. These are found to be powerful methods in domains where analytic solutions have not been proved to be effective. The BFO [5.15] is one such evolutionary computing approach which is based on the foraging behaviour of *E. coli* bacteria in our intestine. In this case foraging is considered as an optimization process in which the bacterium tries to maximize the collected energy per unit foraging time. The BFO has been successfully applied to many real world problems like harmonic estimation [5.16], transmission loss reduction [5.17], active power filter for load compensation [5.18], power network [5.19], load forecasting [5.20], independent component analysis [5.21], identification of nonlinear dynamic systems [5.22-5.23], stock market prediction [5.24] and adaptive channel equalization [5.25].

The basics of PSO algorithm is dealt in Section 2.5.2. The BFO and PSO are derivative free optimization tools in the sense that they do not need the computation of derivatives during training of the weights of the adaptive structure and therefore the solution is less likely to be trapped to local minima. On the other hand the least mean square (LMS) and the recursive least square (RLS) algorithms calculate the slope of the error surface at a current position in all directions, but moves in the direction of the most negative slope. Such optimization methods work satisfactorily when the error surface contains no local minima. But most of the real life problems are multimodal and also are distorted due to additive noise. In case of BFO there are number of parameters which are combinedly used for searching the total solution space. As a result the possibility of avoiding the local minima is higher. The distinct advantages of the BFO and PSO have motivated many researchers to use these tools for identification of complex nonlinear and dynamic systems. The connecting weights of the FLANN model are updated using BFO and PSO techniques instead of using derivative based algorithm. To facilitate the development of the new models efficient BFO and PSO based identification algorithms are proposed in this chapter.

5.2 Dynamic system identification of nonlinear system

The basic principle of system identification is discussed in and depicted in Fig. 3.1. This section also deals with four different identification models and the associated difference equations given in (3.3) - (3.6). In this chapter single-input single-output (SISO) and multi-input multi-output (MIMO) plants of four different nonlinear models are considered. The nonlinear functions $f(\cdot)$ and $g(.)$ associated with the plant are implemented using FLANN-BP rule, FLANN-BFO and FLANN-PSO structures. It is assumed that the plant under consideration is bounded-input-bounded-output (BIBO) stable. In order to achieve the stability and to ensure that the parameters of the ANN model to converge a series-parallel scheme is employed. In this scheme the output of the plant instead of that of the ANN models is fed back to the models during the training operation [5.4].

5.3 A generalized FLANN Structure based identification model

A generalized adaptive identification model of a complex dynamic nonlinear plant is shown in Fig. 5.1. The output of the model $\hat{y}(k+1)$ at $(k+1)th$ instant is given by

$$\hat{y}(k+1) = N_1[x(k)] + N_2[y(k)] \tag{5.1}$$

where N_1 and N_2 represent the low complexity FLANN structures of feed forward and feed back paths respectively. The weights of these structures are updated using PSO and BFO algorithms. Using functional expansion block-1, the input $x(k)$ is nonlinearly expanded as

$$u(k) = [1, x(k), \sin\{\pi x(k)\}, \cos\{\pi x(k)\}.....\sin\{n\pi x(k)\}, \cos\{n\pi x(k)\}]^T \tag{5.2}$$

$$= [u_0(k), u_1(k), ...u_{2n+1}(k)]^T \tag{5.3}$$

There are n number of sine and equal number of cosine expansions of the input sample. The first term $u_0(k)$ is an unity input. Hence altogether there are $(2n+1)$ number of terms in the input vector. Let the weight vector corresponding to the kth input vector defined in (5.3) is given by

$$w(k) = [w_0(k), w_1(k), w_2(k), ..., w_{2n+1}(k)]^T \tag{5.4}$$

The estimated output of the feed forward path is thus given by

$$\hat{y}_1(k+1) = \underline{u}^T(k)\ \underline{w}(k) \tag{5.5}$$

In the similar way, the estimated output of the feedback path is computed as

$$\hat{y}_2(k+1) = \underline{v}^T(k)\ \underline{h}(k) \tag{5.6}$$

$$\text{where } v(k) = [v_0(k), v_1(k), ..., v_{2m+1}(k)]^T \tag{5.7}$$

$$= [1,\ y(k),\ \sin\{\pi y(k)\},\ \cos\{\pi y(k)\}, ...\sin\{n\pi y(k)\},\ \cos\{n\pi y(k)\}]^T \tag{5.8}$$

Here $v_0(k) = 1$.

The net estimated output , $\hat{y}(k+1)$ of the model is given by

$$\hat{y}(k+1) = \hat{y}_1(k+1) + \hat{y}_2(k+1) \tag{5.9}$$

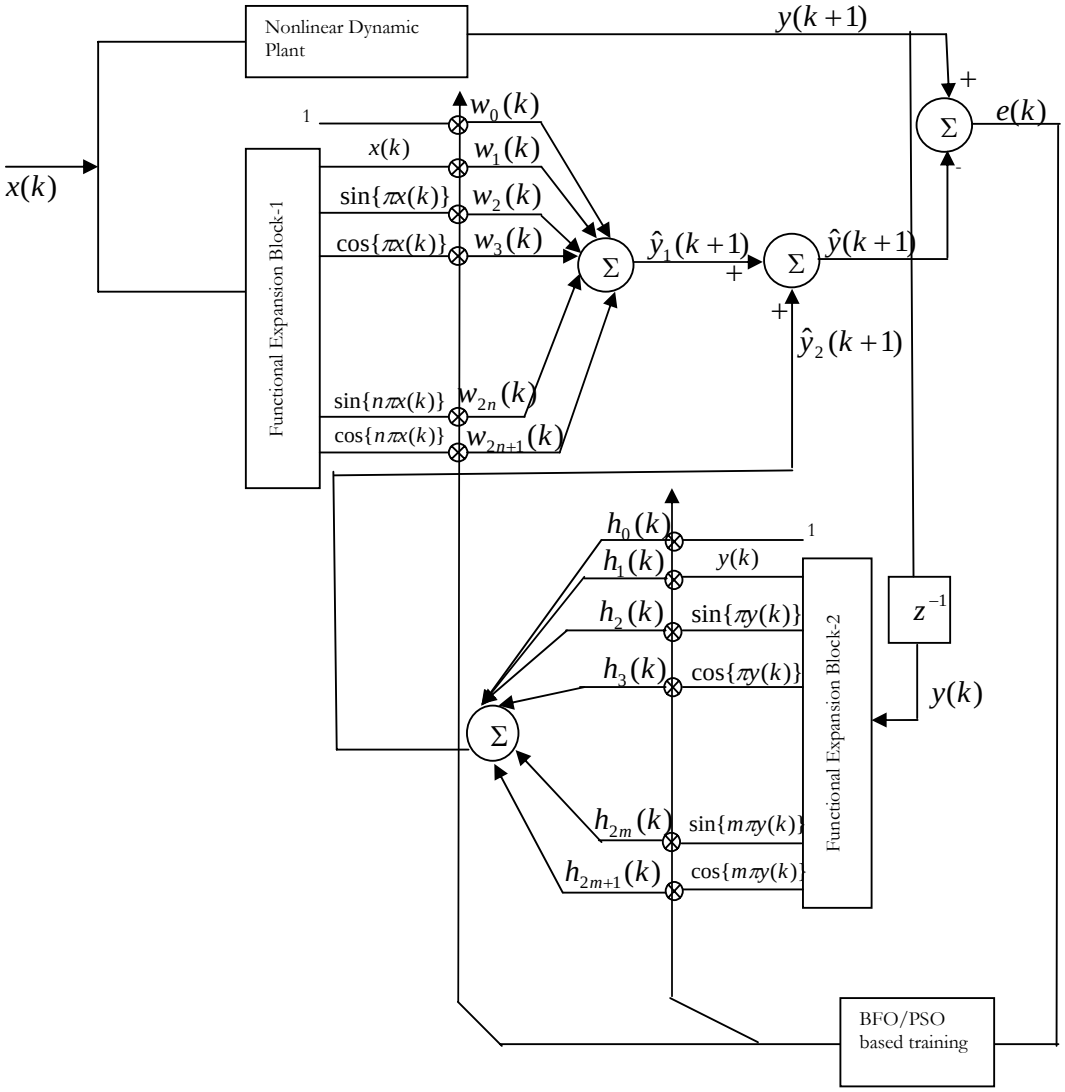


Fig. 5.1 A generalized adaptive model of a complex dynamic nonlinear plant

5.4 BFO and PSO based nonlinear system identification

BFO based identification algorithm

The steps involved in BFO based identification algorithm is presented here.

Step -1 Initialization of parametrs used

(i) S_b = No. of bacteria to be used for searching the total region

(ii) N_{is} = Number of input sample

(iii) p = Number of parameters of the FLANN model to be optimized

(iv) N_s = Swimming length after which tumbling of bacteria is undertaken in a chemotactic loop.

(v) N_c = Number of iterations to be undertaken in a chemotactic loop. Always $N_c > N_s$.

(vi) N_{re} = Maximum number of reproduction to be undertaken

(vii) N_{ed} = Maximum number of elimination and dispersal events to be imposed over the bacteria.

(viii) P_{ed} = Probability with which the elimination and dispersal operation continues.

(ix) The location of each bacterium $\theta(1-p,1-S_b,1)$ is specified by random numbers between [0,1]

(x) The runlength unit $C(i)$ of i th bacterium is assumed to be constant for all bacteria.

Step-2 Generation of desired signal for training

(i) An uniformly distributed random signal over the interval [-1, 1] is generated and simultaneously fed to nonlinear dynamic plant and to the adaptive model which to be trained by BFO algoithm. A series-parallel identification scheme is used for achieving stability during training [5.4].

(ii) The output of the nonlinear plant acts as the desired signal for training.

Step -3 Iterative Identification Algorithm

In this step the bacterial population, chemotaxis, reproduction, elimination and dispersal operations are performed to train the weights of the model.

Initially $j = n = l = 0$

(i) Elimination dispersal loop $l = l + 1$

(ii) Reproduction loop $n = n + 1$

(iii) Chemotaxis loop $j = j + 1$

(a) For $i = 1,2,.....S_b$, the cost function, (in this case mean squared error) $J(i, j, n, l)$ for each i th bacterium is calculated as follows :

(1) N_{is} signal samples are passed through the model.

(2)The output is then compared with the corresponding desired signal to calculate the error.

(3)The sum of squared error averaged over N_{is} is finally stored in $J(i, j, n, l)$. The cost function of the model is calculated for N_{is} input samples as

$$J = \frac{1}{N_{is}} \sum_{k=1}^{N_{is}} e^2(k) \tag{5.10}$$

where

$e(k) = y(k) - \hat{y}(k)$

(4)End of For loop.

(b)For $i = 1,2,.....S_b$, the tumbling/swimming decision is taken.

Tumble: A random vector $\Delta(i),$ with its element $\Delta_m(i),$ $m = 1,2,.....p,$ is computed where each element is a random number in the range [-1, 1].

Move: The move operation is implemented as

$$\theta^i(j+1,n,l) = \theta^i(j,n,l) + C(i) \times \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \tag{5.11}$$

The second term results in an adaptable step size in the direction of tumble for bacterium i.

The new cost function $J(i, j+1, n, l)$ is computed corresponding to the new location of the bacteria.

Swim – (i) Let c =0; (counter for swim length)

(ii) While $c < N_s$ (have not climbed down too long)

Let $c = c + 1$

If $J(j) < J(j-1)$ then by using (5.10) the new cost function $J(i, j+1, n, l)$ is computed

else

let $c = N_s$. This is the end of while statement.

(c)The operation of next bacterium is processed if $i \neq S_b$

(d)If $\min(J)$ {minimum value of J achieved by all the bacteria} is less than the tolerance limit specified then all the loops are broken.

Step 4. If $j < N_c$, the chemotaxis loop beginning from (iii) is continued.

Step 5. Reproduction

(a) For given n and $l,$ and for each $i = 1,2,.....S_b$ let J be the health of i th bacterium. The bacteria are sorted in ascending order of cost functions J (higher cost means lower health).

(b) One half $(S_r = S_b / 2)$bacteria with higher J values die and the remaining S_r bacteria providing minimum MSE values split and are placed at the same location as their parents.

Step 6. If $k < N_{re}$ the reproduction loop from (ii) is continued.

Step 7. Elimination –Dispersal

Bacteria are eliminated and dispersed with probability P_{ed} . This is achieved by eliminating a bacterium and dispersing it to a random location. Same numbers of new bacteria with random locations are added. This process keeps the number of bacteria in the population constant.

In the present study swarming operation is not used to keep the algorithm simple and simultaneously sacrificing little accuracy in the identification task.

PSO based nonlinear system identification

The updating of the weights of the PSO based model is carried out using the training rule as outlined in the following steps:

Step 1. K (K≥500) samples of uniformly distributed random signal in the interval [-1, 1] are generated and simultaneously fed to actual nonlinear system and the adaptive model. A series-parallel identification scheme is used.

Step 2. The output of the plant provides the desired signal. Hence K numbers of desired samples and K numbers of estimated outputs using (5.9) are produced by feeding all the K input samples.

Step 3 . Each of the desired output is compared with the corresponding model output and K errors are produced.

Step 4 . The mean square error (MSE) for a given plant (corresponding to i^{th} particle) is determined by using the relation.

$MSE(i) = \frac{\sum_{k=1}^{K} e_k^2}{K}$ This is repeated for I times.

Step 5. Since the objective is to minimize MSE (i), i = 1 to I the PSO based optimization method is used.

Step 6. The velocity and position of each particle is updated using (4.1) and (4.2) respectively.

Step 7. In each iteration the minimum MSE, (MMSE) which shows the learning behavior of adaptive model from iteration to iteration is stored.
Step 8. When the MMSE has reached the pre-specified level, the optimization process is stopped.
Step 9. At this step the particles attain the same position, which represents the desired solution i. e. the estimated coefficients of the given dynamic plant.

Primarily the identification problem is a optimization problem in the sense that the average mean squared error is to be iteratively minimized. The popular population based optimization algorithms are BFO and PSO. The algorithms have been selected to be used to change the parameters of the identification model in such a way that the squared error fitness function is minimized. This section has dealt the BFO and PSO based nonlinear system identification problem to be used in the simulation study.

5.5 Simulation study

In this section simulation study is carried out to assess the performance of the proposed models when nonlinear identification of static and dynamic plants described by (3.3)-(3.6). In these examples, the series-parallel model is used to identify these plants and BFO and PSO algorithms are used to train the connecting weights of the FLANN structure of the models. The performance of the proposed (FLANN-BFO and FLANN-PSO) approaches is obtained from simulation and compared with that obtained by FLANN-BP method [5.12]. For training the weights of FLANN-BP model, 50,000 iterations are carried out by using an uniformly distributed random signal over the interval [-1,1] as input. During the test phase, the effectiveness of the proposed models are studied by using the parallel scheme where the input to the identified model used is given by

$$x(k)=\begin{cases}\sin\dfrac{2\pi k}{250} & for\ k\le 250\\ 0.8\sin\dfrac{2\pi k}{250}+0.2\sin\dfrac{2\pi k}{25} & for\ k>250\end{cases} \tag{5.12}$$

A quantitative measure for performance evaluation used is the normalized mean square error (NMSE) defined in [5.26] as

$$NMSE = \frac{1}{\sigma^2 T_D}\sum_{k=1}^{T_D}[y(k)-\hat{y}(k)]^2 \tag{5.13}$$

where $y(k)$ and $\hat{y}(k)$ represent the plant and model outputs at k th discrete time, respectively and σ^2 denotes variance of the plant output sequence over the test duration T_D.

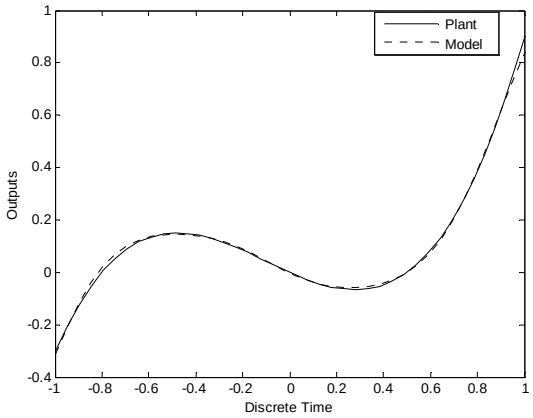
Static Systems

Two examples of static systems [5.4, 5.12] used in the simulation study are

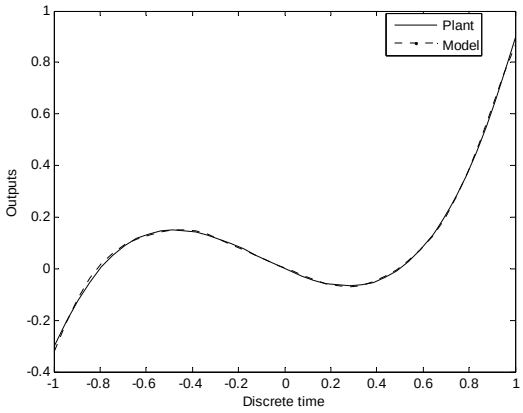
Example 1 : $f_1(x) = x^3 + 0.3x^2 - 0.4x$

Example 2 : $f_2(x) = 0.6\sin(\pi x) + 0.3\sin(3\pi x) + 0.1\sin(5\pi x)$

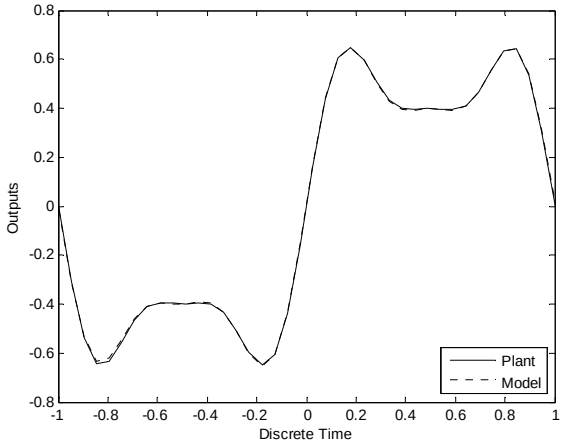
In case of FLANN-BFO and FLANN-PSO nine input nodes for first example and eleven input nodes for second example are used to obtain the best possible identification results. The number of connecting weights including the threshold is ten and twelve respectively. These weights are updated using the bacterial foraging or particle swarm optimization algorithm. But in case of FLANN-BP, fifteen input nodes including a bias input are used to achieve similar performance. In both cases the input pattern is expanded using trigonometric expansion and the nonlinearity used is $\tanh(.)$ function. The convergence coefficient is set to 0.1 in case of FLANN-BP where as the parameters used for FLANN-BFO are : S_b = 16, N_{is} = 100, p =10, N_s =3, N_c =5, N_{re} =100-130, N_{ed} =5, P_{ed} = 0.25, $C(i)$ = 0.0075. Similarly the parameters used in case of PSO based simulation are no. of particles=30, no. of input samples =200, $c_1 = c_2 = 1.042, v_{\max} = 1$. The results of identification of Examples 1 and 2 are shown in Figs. 5.2(a) – (d). From these results it is clear that the BFO and PSO based FLANN models provide excellent agreement between plant and model responses. Using the same number of expansions in both the examples the estimation error provided by the FLANN-BFO and FLANN-PSO are found (Table 5.1) to be lower than that of FLANN-BP approach.



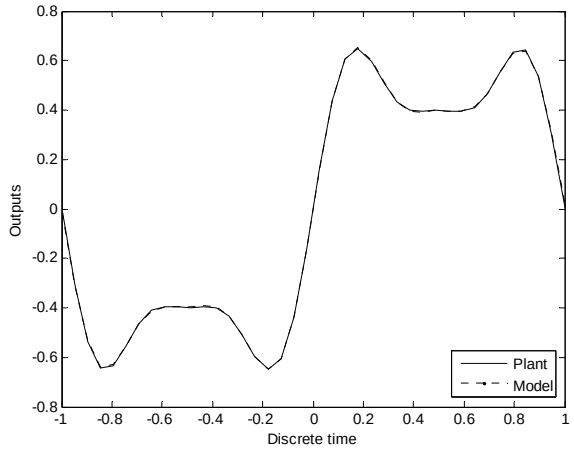
(a) FLANN-BFO (nine expansions)



(b) FLANN-PSO (nine expansions)



(c) FLANN-BFO (eleven expansions)



(d) FLANN-PSO (eleven expansions)

Fig. 5.2 Response matching of static systems ((a), (b) for Example 1 and (c) and (d) for Example 2)

Dynamic (SISO) Systems

Example 3: The difference equation of the plant [5.4, 5.12] to be identified is given as

$$y(k+1) = 0.3y(k) + 0.6y(k-1) + g[x(k)] \tag{5.14}$$

The linear parameters are 0.3 and 0.6 and the unknown nonlinear functions $g_i(.)$ are given by

$$g_1(x) = \frac{4.0x^3 - 1.2x^2 - 3.0x + 1.2}{0.4x^5 + 0.8x^4 - 1.2x^3 + 0.2x^2 - 3.0} \tag{5.15}$$

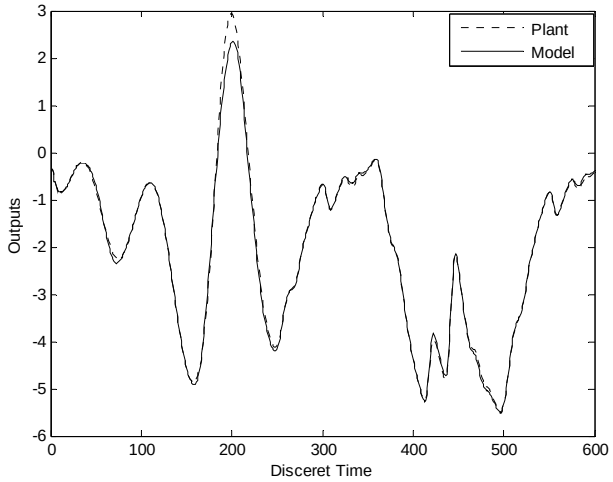
$$g_2(x) = 0.5\sin^3(\pi x) - \frac{2.0}{x^3 + 2.0} - 0.1\cos(4\pi x) + 1.125 \tag{5.16}$$

To identify the plant, a series-parallel model is used whose difference equation is given as

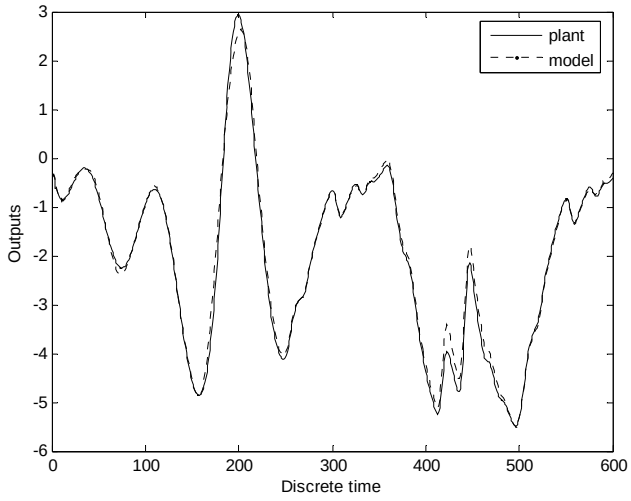
$$\hat{y}(k+1) = 0.3y(k) + 0.6y(k-1) + \mathrm{N}[x(k)] \tag{5.17}$$

where $\mathrm{N}[\mathrm{x}(\mathrm{k})]$ represents either the FLANN-BP, FLANN-BFO or FLANN-PSO model. The FLANN input is expanded to nine terms by using trigonometric expansion and BFO or PSO algorithm is used to update its connecting weights. The parameters used for BFO based FLANN model are same as used in Example 1 except that N_{re}=60. The parameters used for PSO are : no. of particles=30, no. of input samples=200, $c_1 = c_2 = 1.042$ *and* $v_{\max} = 1$. In case of FLANN-BP the input is expanded to fourteen trigonometric terms and delta rule is used to train the weights. Both convergence

parameter, μ and momentum parameter, η are chosen to be 0.1. The results of identification of (5.14) with nonlinear functions defined in (5.15) and (5.16) are shown in Figs. 5.3 (a) and (b) and Figs. 5.4(a) and (b) respectively.

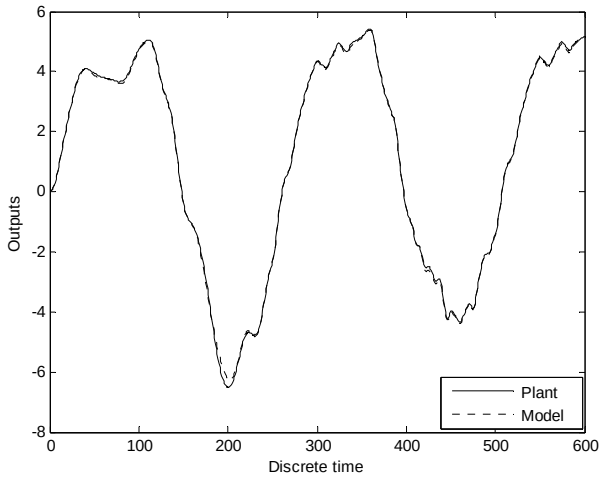


(a) Using FLANN-BFO (nine expansions)

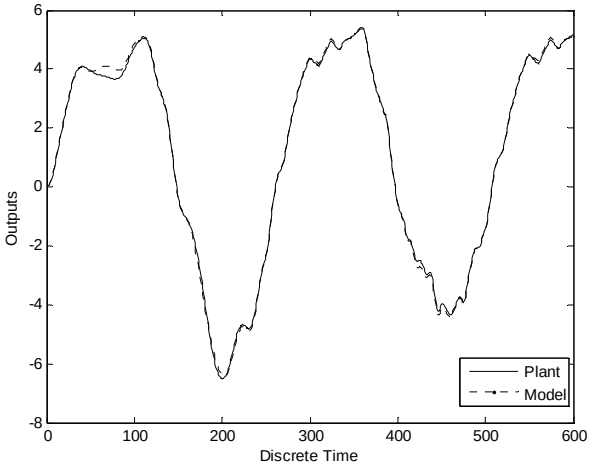


(b) Using FLANN-PSO (nine expansions)

Fig. 5.3 Comparison of response of the dynamic plant of Example 3 using nonlinearity defined in (5.15)



(a) Using FLANN-BFO (nine expansions)



(b) Using FLANN-PSO (nine expansions)

Fig. 5.4 Comparison of response of the dynamic plant of Example 3 using nonlinearity defined in (5.16)

From these results it is evident that the FLANN-BFO and FLANN-PSO methods provide accurate identification performance. Further, from Table 5.1 it is observed that the BFO and PSO based approches yield lower NMSE compared to its FLANN-BP counterpart.

Example 4 : In this example the plant [5.4, 5.12] to be identified is of Model-2 type and is represented by the difference equation

$$y(k+1) = f[y(k), y(k-1)] + x(k) \tag{5.18}$$

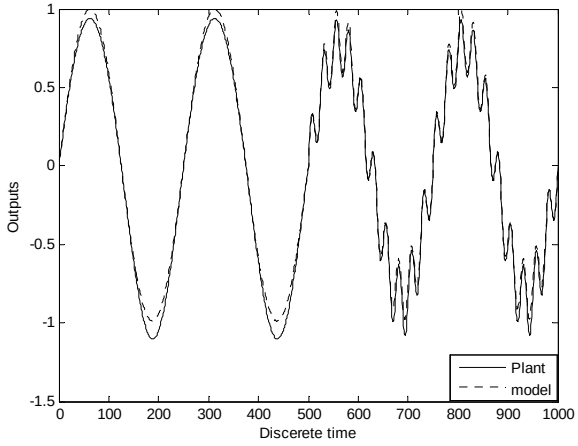
The unknown nonlinear function f is given by

$$f(y_1, y_2) = \frac{y_1 y_2 (y_1 + 2.5)(y_1 - 1.0)}{1.0 + y_1^2 + y_2^2} \tag{5.19}$$

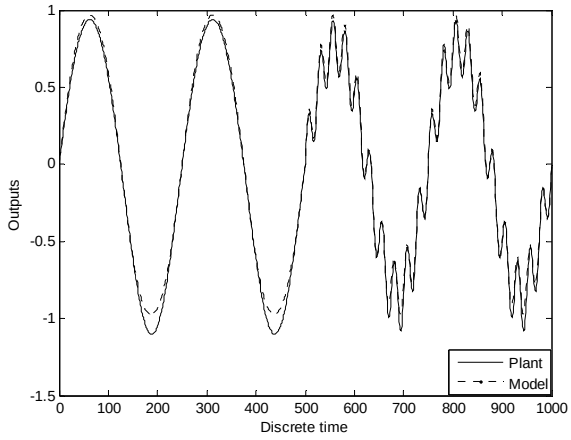
In this case the series-parallel scheme of the model used is given by

$$\hat{y}(k+1) = \mathrm{N}[(y(k), y(k-1)] + x(k) \tag{5.20}$$

The symbol N represents the model defined in example 4. In FLANN-BFO and FLANN-PSO, each of the two inputs are expanded to six terms each and the BFO or PSO is used to train the weights. The parameters used for BFO based FLANN model are S_b = 16, N_{is} = 100, p =13, N_s =3, N_c =5, N_{re} =240, N_{ed} =5, P_{ed} = 0.25 and $C(i)$ = 0.0075. For PSO the parameters used are no. of particles=30, no. of input samples =500, $c_1 = c_2 = 1.042$ *and* $v_{\max} = 1$.In case of FLANN-BP the two inputs are expanded into 24 terms and the convergence and the momentum parameters are set at 0.05 and 0.1 respectively. The response obtained from the plant and the two models are shown in Figs. 5.5(a) and (b). In this case also it is observed that the response matching of FLANN-BFO and FLANN-PSO is excellent. Results presented in Table 5.1 also indicate improved performance of the two new models compared to the results of FLANN-BP model.



(a) Using FLANN-BFO (twelve expansions)



(c) Using FLANN-PSO (twelve expansions)

Fig. 5.5 Comparison of response of the dynamic plant of Example 4

Example 5: In this case the plant is of Model-3 type and is described by the difference equation

$$y(k+1) = f[y(k)] + g[x(k)] \tag{5.21}$$

where the unknown nonlinear functions $f(.)$ and $g(.)$ are represented as

$$f(y) = \frac{y(y+0.3)}{1.0+y^2} \tag{5.22}$$

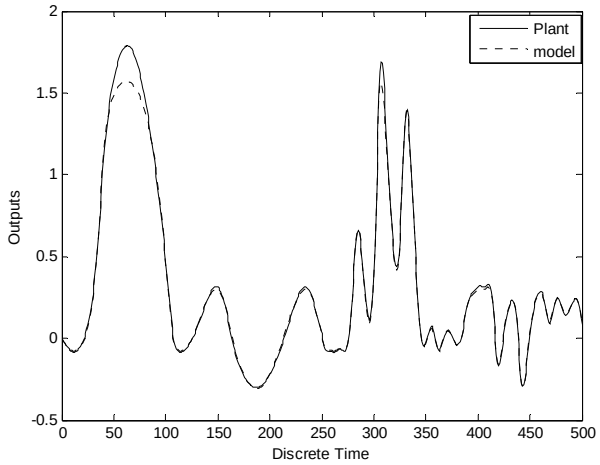
$$g(x) = x(x+0.8)(x-0.5) \tag{5.23}$$

The model is represented by a series-parallel scheme

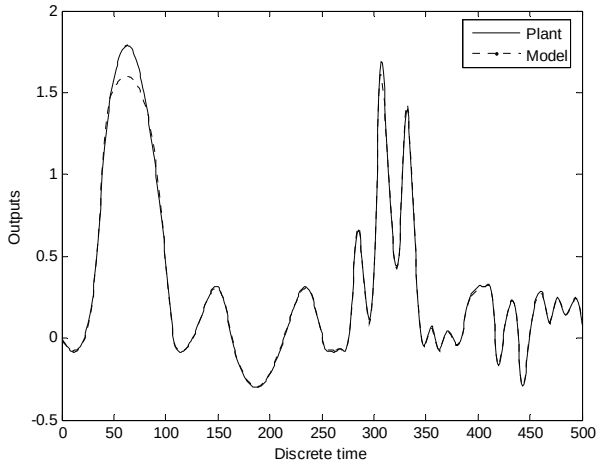
$$\hat{y}(k+1) = \mathrm{N}_1[y(k)] + \mathrm{N}_2[x(k)] \tag{5.24}$$

where N_1 and N_2 represent the FLANN-BP, FLANN-BFO or FLANN-PSO model. In FLANN-BP model N_1 and N_2 structures contain 14 and 24 trigonometric expansions respectively whereas in case of FLANN-BFO and FLANN-PSO seven and five number of expansions are used. The convergence and momentum parameters are chosen to be 0.1 in case of FLANN-BP model. The parameters used for BFO based FLANN model are S_b = 16, N_{is} = 100, p =14, N_s =3, N_c =5, N_{re} =120, N_{ed} =5, P_{ed} = 0.25 and $C(i)$ = 0.0075. For PSO, the parameters used are no. of particles=30, no. of input samples =500, $c_1 = c_2 = 1.042, v_{\max} = 1$. The responses obtained from the plant and various models are

compared in Figs. 5.6(a) and (b) and the computed NMSE is presented in Table 5.1. These results also indicate superior performance of the proposed technique over its FLANN-BP counterpart.



(a) Using FLANN-BFO (twelve expansions)



(b) Using FLANN-PSO (twelve expansions)

Fig. 5.6 Comparison of response of the dynamic plant of Example 5

Example 6 : MIMO System

The two input and two output nonlinear disceret time plant [5.14] is described by

$$\begin{bmatrix} y_1(k+1) \\ y_2(k+1) \end{bmatrix} = \begin{bmatrix} \dfrac{y_2(k)}{1+y_1^2(k)} \\ \dfrac{y_1(k)}{1+y_1^2(k)} \end{bmatrix} + \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} n_1(k) \\ n_2(k) \end{bmatrix} \tag{5.25}$$

where $n_1(k)$ and $n_2(k)$ are white Guassian noise with zero mean and covariance of 0.0009.

The estimated outputs are given by

$$\begin{aligned} \hat{y}_1(k+1) &= f_1[y_1(k), y_2(k), x_1(k), x_2(k)] \\ \hat{y}_2(k+1) &= f_2[y_1(k), y_2(k), x_1(k), x_2(k)] \end{aligned} \tag{5.26}$$

The inputs $x_1(k)$ and $x_2(k)$ are

$$x_1(k) = \cos\left(\frac{2\pi k}{100}\right)$$

$$x_2(k) = \sin\left(\frac{2\pi k}{100}\right) \tag{5.27}$$

The block diagram for the identification of the MIMO plant (5.25) is given in Fig. 5.7.

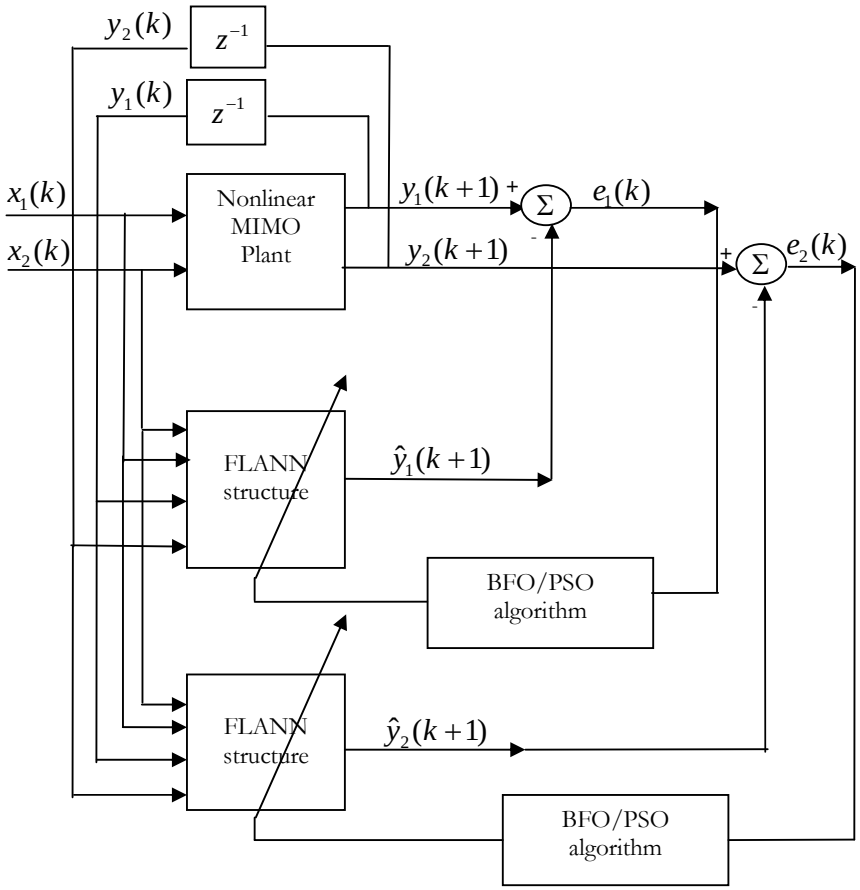
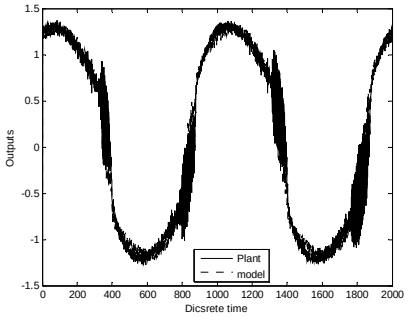
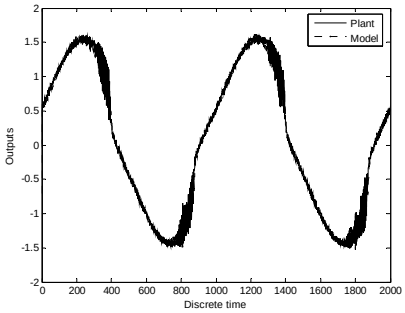


Fig. 5.7 Block diagram of nonlinear MIMO plant identification

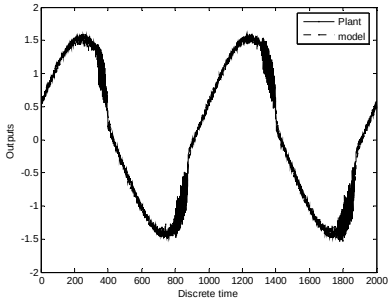
In case of FLANN-BP model, each of the inputs $y_1(k), y_2(k), x_1(k)$ *and* $x_2(k)$ are expanded into three terms each using Chebyshev polynomials [5.13]. The weights are updated using delta rule. A parallel scheme is used for the identification purpose. The actual and model responses of the MIMO system are displayed in Figs.5.8 (a) and (b) for FLANN-BFO and Figs. 5.8 (c) and (d) for FLANN-PSO. The proposed methods show improved agreement between the estimated and true responses. However the FLANN-BP method shows poor response matching capability as evident fron Figs. 5.8 (e) and (f).



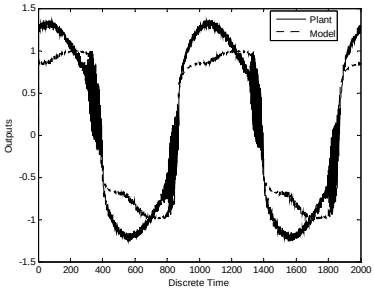
(a) First output using FLANN-BFO



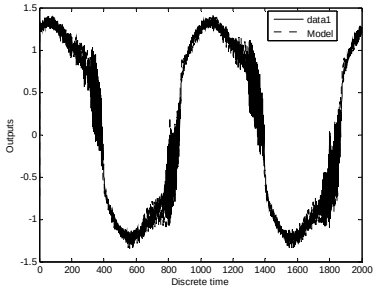
(d) Second output using FLANN-PSO



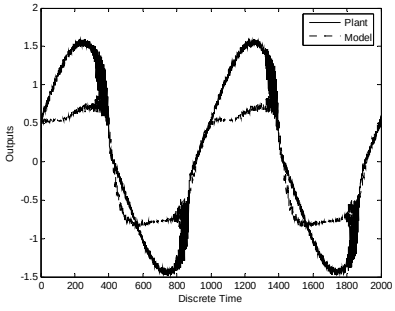
(b) Second output using FLANN-BFO



(e)First output using FLANN-BP



(c) First output using FLANN-PSO



(f) Second output using FLANN-BP

Fig. 5.8 Response matching of MIMO system of Example 6

Table 5.1

Comparison of NMSE(dB) computed for different examples of two different models

Example No.	For higher input expansion		For same number of input expansions			
	FLANN-BP		No. of expansions	FLANN-BP	FLANN-BFO	FLANN-PSO
	NMSE(dB)	No. of exapnsions				
Ex-1	-23.21	14	09	-20.66	-27.30	-29.77
Ex-2	-34.19	14	11	-30.71	-38.98	-42.65
Ex-3with (5.15)	-25.48	14	09	-18.51	-27.09	-20.90
Ex-3with (5.16)	-32. 82	14	09	-28.16	-33.77	-31.02
Ex-4	-21.01	24	12	-20.57	-22.75	-22.88
Ex-5	-19.24	38	12	-18.29	-21.09	-22.58

Table 5.2
Comparison of Computational Complexities of various system identification models

Types of models	No. of tanh ( )	No. of Cos/Sin	No. of weights	No. of Adds.	No. of Muls.
Example-1					
FLANN-BP	1	14	15	14	15
FLANN-BFO/ FLANN-PSO	0	09	10	9	10
Example-2					
FLANN- BP	1	14	15	14	15
FLANN-BFO/ FLANN-PSO	0	11	12	11	12
Example-3					
FLANN- BP	1	14	15	14	15
FLANN-BFO/ FLANN-PSO	0	09	10	9	10
Example-4					
FLANN- BP	1	24	25	24	25
FLANN-BFO/ FLANN-PSO	0	12	13	12	13
Example-5					
FLANN-BP	2	38	40	39	40
FLANN-BFO/ FLANN-PSO	0	12	14	13	14

Extensive simulation studies exhibit that the FLANN structure with BFO and PSO based training models provide better output response as well as require substantially lesser number (about 200 to 600) of iterations to converge compared to 50,000 iterations required by FLANN-BP model. In BFO based training, only 100 input samples and 16 number of bacteria population are required for convergence whereas in case of PSO 200-500 input

samples and 30 particles are required. The computational requirements involved per iteration of the algorithm both in the proposed and FLANN-BP methods are evaluated and listed in Table 5.2. This comparison also indicates that the FLANN-BFO and FLANN-PSO methods involve lesser operations compared to FLANN-BP in all the examples simulated. As shown in Table 5.1, the NMSE obtained in each example by the proposed FLANN-BFO and FLANN-PSO models is also less in comparison to the FLANN-BP approach.

5.6 Conclusion

This Chapter introduces the problem and importance of adaptive nonlinear system identification. Then the shortcomings of the conventional identification methods are highlighted. Two new approaches based on swarm intelligence are proposed to identify complex nonlinear dynamic plants. The corresponding BFO and PSO based identification algorithms are presented in this chapter. The performance of the proposed methods are assessed by simulating various standard nonlinear dynamic and MIMO systems. These results have also been compared with those obtained by FLANN-BP based approach. The comparison reveals that the new methods of identification are fast, more accurate and involve less computation compared to its FLANN-BP counterpart. Thus the proposed new approaches are promising methods of achieving efficient nonlinear dynamic system identification.

References

[5.1] S. Haykin , Neural Networks, Ottawa, ON Canada: Maxwell Macmillan, 1994.

[5.2] P. S. Sastry, G. Santharam and K. P. Unnikrishnan, "Memory neural networks for identification and control of dynamical systems", IEEE Trans. Neural Networks, vol. 5, pp. 306-319, 1994.

[5.3] A. G. Parlos., K. T. Chong and A. F. Atiya, "Application of recurrent multilayer perceptron in modeling of complex process dynamics", IEEE Trans. Neural Networks, vol. 5, pp. 255-266, 1994.

[5.4] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", IEEE Trans. on neural networks, vol. 1, no. 1, pp. 4-27, 1990.

[5.5] D. H. Nguyen and B. Widrow, "Neural networks for self-learning control system", Int. J. Conttr., vol. 54, no. 6, pp. 1439-1451, 1991.

[5.6] G. Cembrano, G. Wells, J. Sarda and A. Ruggeri, "Dynamic control of a robot arm based on neural networks", Contr. Eng. Practice, vol 5, no. 4, pp. 485-492, 1997.

[5.7] T. Poggio and F. Girosi, "Networks for approximation and learning", Proceeding of IEEE, vol. 78, no. 9, pp. 1481-1497, 1990.

[5.8] S. Chen, S. A. Billings and P. M. Grant, "Recursive hybrid algorithm for nonlinear system identification using radial basis function networks", Int. J. Contr., vol. 55, no. 5, pp. 1051-1070, 1992.

[5.9] S. V. T. Elanayar and Y. C. Shin, "Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems", IEEE Trans. Neural Network, vol. 5, pp. 594-603, 1994.

[5.10] Q. Zhang and A. Benveniste, "Wavelet networks", IEEE Trans. Neural Networks, vol. 3, pp. 889-898, 1992.

[5.11] Y. C. Pati and P. S. Krishnaprasad, "Analysis and synthesis of feed forward neural networks using discrete affine wavelet transforms", IEEE Trans. Neural Networks, vol. 4, pp. 73-85, 1993.

[5.12] J. C. Patra, R. N. Pal, B. N. Chatterji and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks", IEEE Trans. in Systems, Man and Cybernetics-Part B: Cybernetics, vol. 29, no. 2, pp. 254-262, 1999.

[5.13] J. C. Patra and A. C. Kot, "Nonlinear dynamic system identification using Chebyshev functional link artificial neural networks", IEEE Trans. in Systems, Man and Cybernetics-Part B: Cybernetics, vol. 32, no. 4, pp. 505-511, 2002.

[5.14] S. Purwar, I. N.Kar and A. N. Jha, "Online system identification of complex systems using Chebyshev neural networks ", Applied Soft Computing, Elsevier, pp. 364-372, 2007.

[5.15] K. M. Passino, "Biomimicry of Bacterial Foraging for distributed optimization and control", IEEE control system magazine, vol. 22, no. 3, pp. 52-67, 2002.

[5.16] S. Mishra, "Hybrid least square adaptive bacterial foraging strategy for harmonic estimation", IEE Proc. on Gener., Transm., Distrib., vol. 152, no. 3, pp. 379-389, 2005.

[5.17] M. Tripathy, S. Mishra., L. L. Lai and Q. P. Zhang, "Transmission loss reduction based on FACTS and Bacteria Foraging algorithm", PPSN, pp. 222-231, 2006.

[5.18] S. Mishra and C. N. Bhende, "Bacterial Foraging technique based optimized active power filter for load compensation", IEEE Trans. on Power Delivery, vol. 22, no.1, pp. 457-465, 2007.

[5.19] M. Tripathy and S. Mishra, "Bacteria foraging based to optimize both real power loss and voltage stability limit", IEEE Trans. on Power Systems, vol. 22, no. 1, pp. 240-248, 2007.

[5.20] L. Ulagammai., P. Venkatesh, S. P. Kannan and N. P. Padhy, "Application of bacteria foraging technique trained and artificial and wavelet neural networks in load forecasting". Neurocomputing, pp. 2659-2667, 2007.

[5.21] D. P. Acharya, G. Panda, S. Mishra and Y. V. S. Lakhshmi, "Bacteria foraging based independent component analysis". Proc. of Int. Conf. on Computational Intelligence and Multimedia Applications, vol. 2, pp. 527-531, 2007.

[5.22] Babita Majhi and G. Panda**,** "Bacterial Foraging based Identification of Nonlinear Dynamics System", Proc. of IEEE International Congress on Evolutionary Computation(CEC-2007), Singapore, 25-28, September, 2007, pp.1636-1641.

[5.23] G. Panda, Babita Majhi and S. Mishra, "Nonlinear System Identification using Bacterial Foraging based Learning", Proc. of IET 3rd International Conference on Artificial Intelligence in Engineering Technology (ICAIET-2006), Kota kinabalu, Malaysia, 22-24, November, 2006, pp. 120-125.

[5.24] R. Majhi, G. Panda, G. Sahoo and D.P. Das, "Stock market prediction of S&P 500 and DJIA using Bacterial Foraging Optimization technique", IEEE Congress on Evolutionary Computation (CEC 2007), Singapore, 25-28 September 2008, pp.2569-2575.

[5.25] Babita Majhi, G. Panda and A. Choubey**, "**On The Development of a new Adaptive Channel Equalizer using Bacterial Foraging Optimization Technique", Proc. of IEEE Annual India Conference (INDICON-2006), New Delhi, India, 15th-17th September, 2006, pp. 1-6

[5.26] N. A. Gershenfeld. and A. S. Weigend, "The future of time series: Learning and understanding", Time Series Prediction: Forecasting the future and past, Reading, MA: Addison-Wesley, pp. 1-70, 1993.

Chapter

6

Robust Identification and Prediction using Particle Swarm Optimization Technique

6.1 Introduction

THE objective of identification is to determine a suitable mathematical model of a given system/process, useful for predicting the behavior of the system under different operating conditions. Its another objective is to design a controller which allows the system to perform in a desired manner. Most of the practical plants and systems are nonlinear and dynamic in nature and hence identification of such complex plants is a challenging task. Accurate and fast identification of real time nonlinear complex processes is still a difficult problem. Further, in many practical situations, building of a proper model of a plant becomes difficult when outliers are present or few data are missing from the output samples of the plant or the training signal. Under such adverse conditions the training of models becomes ineffective when conventional learning algorithms such as the least mean square (LMS) or recursive least square (RLS) type algorithms are used. But all these derivative based algorithms have been

derived by minimizing the square of the error as the cost function. In recent past many bioinspired and evolutionary computing tools such as genetic algorithm (GA), particle swarm optimization (PSO), bacterial foraging optimization (BFO) and ant colony optimization (ACO) have been reported and have been applied to optimization and identification tasks. In case of the derivative free algorithms conventionally the mean square error (MSE) is used as the fitness or cost function. Use of MSE as cost function leads to improper training of adaptive model when outliers are present in the desired signal. Therefore there is a need for identification of complex plants which are nonlinear and dynamic in nature. It is a fact that the traditional regressors employ least square fit which minimizes the Euclidean norm, while the robust estimator is based on a fit which minimizes another rank based on a norm called Wilcoxon norm [6.1]. It is known in statistics that linear regressors developed using Wilcoxon norm are robust against outliers. Using such norm new robust machines have recently been proposed for approximation of nonlinear functions [6.2]. In the present investigation a new method of robust identification of nonlinear dynamic systems or plants is developed by minimizing robust cost function (RCF) [6.1], [6.44], [6.47] of errors of a functional link artificial neural network (FLANN) model using a derivative free PSO technique. The identification performance of the new method is evaluated through simulation study and is compared with the results obtained from corresponding Euclidean norm based PSO technique. Hence the main contribution of the paper is the formulation of complex identification task as a robust optimization problem of RCF of the FLANN model. The second contribution is the effective minimization of this norm employing a population based derivative free PSO technique which essentially adjusts the connecting weight of feed forward and feed back paths of the model. The third contribution is the selection of appropriate FLANN structure as the backbone of the model which is a single layer ANN structure and offers low complexity. The robust identification performance in presence of outliers in the training signal is then shown through simulation of some bench mark nonlinear dynamic identification problems. Many research papers have been reported in the literature to identify both static and dynamic nonlinear systems. The Artificial Neural Network (ANN) has been employed for many identification and control purpose[6.3-6.5] but at the expense of large computational complexity. Narendra and Parathasarathy [6.6] have used the MLP architecture with back propagation learning algorithm for effective identification and control of dynamic systems [6.7] and robot arm control [6.8]. Similarly the Radial Basis Function (RBF) network has been introduced to develop system identification model of nonlinear dynamic systems [6.9-6.10]. In recent past, the wavelets in place of RBF has been suggested in neural network [6.11-6.12] to

develop efficient identification model. Further, the Functional Link Artificial Neural Network (FLANN), a computationally efficient single layer ANN, has been reported in the literature as an useful alternative to MLP for many applications. This single layer ANN has also been successfully employed for identification of nonlinear systems [6.13]. Recently Chebyshev-FLANN has been proposed for identification of nonlinear dynamic systems [6.14].

The basics of PSO is dealt in Section 2.5.2. In addition to the applications of PSO described in Section 4.2 it is also applied for reactive power and voltage control [6.15, 6.16], economic dispatch [6.17] -[6.20], power system reliability and security [6.21], generation expansion problem [6.22, 6.23], state estimation [6.24, 6.25], controller tuning [6.26, 6.27], system identification and control [6.28, 6.29], capacitor placement [6.30, 6.31], short term load forecasting [6.32], generator contribution to transmission systems [6.33], industrial applications [6.34], task assignment problem [6.35], to solution of Sudoku puzzles [6.36], electromagnetic design [6.37], unit commitment problem [6.38] and optimization of multimodal functions [6.39].

In the past many robust learning algorithms have been proposed for training different adaptive networks. A robust BP learning algorithm that is resistant to the noise effects has been derived in [6.40]. However the convergence of this algorithm is very slow. Another robust learning algorithm has been reported [6.41] for recurrent neural network. This algorithm is based on filtering outliers from data followed by estimating parameters from the filtered data. The new method makes better prediction of electrical demand compared to conventional methods. In a letter [6.42] a robust learning method has been proposed for RBF network and has been applied for function approximation. Kadir Liano has reported [6.43] a mean log squared error (MLSE) cost function (CF) and has shown that the new cost function yields algorithm which is robust compared to conventional squared error based cost function. In another publication the authors have used the fuzzy-neural network and β-spline membership function for function approximation with outliers in training data [6.44] and have shown that their algorithm is more flexible and efficient than the one reported in [6.38]. In 1998 a robust interval regression analysis has been suggested which provides robust performance against outliers as well as improvement in rate of convergence [6.45]. A robust objective function is suggested in [6.46] for RBF networks to reduce the influence of outliers. The authors have shown that the proposed objective function yields better function approximation compared to its least square (LS) counterpart. In [6.47], the authors have proposed robust learning algorithms of fuzzy neural

network to reduce the outlier effects during training. They have tested the robustness of their algorithm through simulation of various function approximation problems. Chuang et al have recently proposed [6.48] a robust TSK fuzzy modeling approach with improved performance for function approximation in presence of outliers. A novel regression approach has recently been reported [6.49] to enhance the robust capability of the support vector regressor. In their approach they have used a cost function which works satisfactorily when maximum 10% outliers are present in the training set. In 2004, a robust analysis of linear models is presented using Wilcoxon norm [6.1] and it has been shown that the proposed norm is more robust to outliers compared to its least square counterpart. Recently Hsieh et al have proposed robust learning rules for neural network, fuzzy neural network and kernel based regressor using a Wilcoxon norm [6.2] which is different from other reported norms and have shown that the new norm-based algorithm exhibit robust better performance when the percentage of outliers is as high as 40%.

6.2 Formulation of PSO based nonlinear system identification model

The identification scheme of a dynamic nonlinear system is shown in Fig. 6.1 in which $x(n)$, $\hat{y}(n)$, $y(n)$ and $e(n)$ denote the input, output of the model, the output of the system and the error between the two at n th time instant respectively. The input $x(n)$ is an uniformly generated white signal. Therefore,

$$e(n) = y(n) - \hat{y}(n) \tag{6.1}$$

In the present study, identification of single-input single-output dynamic (SISO) plants of four different nonlinear models [6.6] described by the difference equations given in (3.3)-(3.6) are considered.

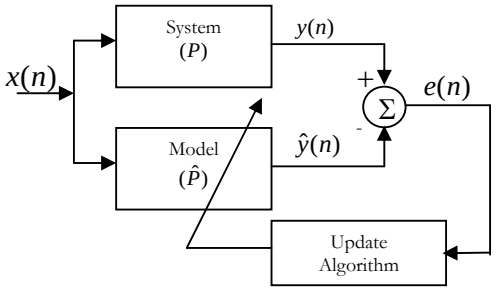


Fig. 6.1 Identification scheme of a dynamic system

The nonlinear functions associated with these plants are implemented using single layer nonlinear FLANN structure and its weights are trained by minimizing three different CFs using PSO algorithm. It is assumed that the plant under consideration is bounded-input-bounded-output (BIBO) stable. In order to achieve the stability and to ensure that the parameters of the FLANN model converge, a series-parallel scheme is employed. In this scheme the output of the plant instead of that of the ANN models is fed back to the models during the training phase [6.6]. Fig. 6.2 depicts a detailed diagram for identification of a nonlinear dynamic plant using a PSO based robust norm minimization technique. The architecture of the proposed adaptive model is taken to be two functional link artificial neural networks (FLANNs) [6.50] as shown in the same figure. The output for n th input sample at k th generation, $y_n(k)$ is used as input to the feedback FLANN structure. In this method N input samples $x(n), (1 \le n \le N)$ uniformly distributed between [-1, 1] are used to develop the model. The same set of input is used for all particles. The output $y_n(k)$ may be computed as

$$y_n(k) = y_n^{'}(k) + \eta(k) \tag{6.2}$$

where $\eta(k)$ represents outliers at randomly selected locations and zero magnitude at remaining samples at k th generation. For developing the proposed FLANN-PSO based adaptive identification model the combined feed forward and feed back weights are considered as one particle. In the beginning, P such particles called a swarm is chosen to represent a population of random solutions. Each weight-particle of the model which is updated using PSO based technique has a random velocity and flies within the solution space. Each particle has also memory to keep track of its previous best position and the corresponding fitness value. Similarly each swarm remembers its best solution achieved so far. The velocity and position of each weight-particle are updated using its personal best position and global best position of the swarm.

The output of the model for p th particle, n th input sample and at k th generation is given by

$$\hat{y}_{n,p}(k) = \underline{\hat{Z}}_n^T(k)\, \underline{W}_p(k) \tag{6.3}$$

where

$$\underline{\hat{Z}}_n(k) = [z_{1,n} \quad z_{2,n} \quad \ldots\ldots\ldots z_{L,n_1} \quad z_{L_1+1,n} \quad \ldots\ldots\ldots z_{L,n}]^T$$

$$\underline{W}_p(k)=[w_{1,p}(k)\ \ w_{2,p}(k).....w_{L_1,p}(k)\ \ w_{L_1+1,p}(k).....w_{L,p}(k)]^T \quad (6.4)$$

$z_{l_1,n}\ (1 \le l_1 \le L_1)$ and $z_{l_2,n}\ (1 \le l_2 \le L - L_1)$ represent the trigonometrically expanded values of the input and output vectors respectively. $\underline{W}_p(k)$ denotes the weight vector of forward and backward structure of the FLANN model . When the n th input sample $x(n)$ is applied the input and output of the shift register contents of Fig.6.2 are given by $[x(n), x(n-1),.....,x(n-T_1+1)]$ and $[y_n(k-1), y_n(k-2),.....y_n(k-T_2)]$ respectively. The symbols T_1 and T_2 represent number of input and output nodes. Each feed forward input sample $x(n)$ is nonlinearly expanded by using trigonometric expansion scheme such as

$[x(n), \sin\{2\pi.x(n)\}, \cos\{2\pi.x(n)\}, \sin\{3.2\pi.x(n)\}, \cos\{3.2\pi.x(n)\}... \sin\{(2s-1).2\pi.x(n)\}, \cos\{(2s-1).2\pi.x(n)\}]$

The symbol s represents the number of sine or cosine expansions of each input sample. The main motivation of employing nonlinear expansions of input samples is to create a nonlinear environment using an adaptive linear combiner which is expected to improve identification of nonlinear dynamic systems. This single layer adaptive architecture effectively substitutes the needs of multilayer artificial neural network. In the same way each feed back sample $y_n(k-1)$ is expanded to same number of trigonometric terms. In the present investigation trigonometric expansion is chosen as it is observed to perform better than the power series based expansions [6.50]. L_1 and L_2 represent the number of expanded terms for feed forward and feed back inputs respectively where $L_1 = T_1 \times (2s+1)$ and $L_2 = T_2 \times (2s+1)$. The total number of expanded terms thus becomes $L = L_1 + L_2$. The error produced at n th input sample, k th generation and for p th particle is given by

$$e_{n,p}(k) = y_n(k) - \hat{y}_{n,p}(k) \quad (6.5)$$

where $y_n(k)$ represents the output of the nonlinear dynamic plant corresponding to n th input sample and at k th generation.

This plant output serves as a training signal for all particles of the proposed model.

In the first part of the investigation, the mean square error defined in (6.6) is computed for each particle and the PSO tool is used to minimize this CF by iteratively changing the weight-particles of the adaptive identification model.

$$E_p(k) = \sum_{n=1}^{N} e_{n,p}^2(k) / N \tag{6.6}$$

for

$p = 1, 2, \ldots\ldots\ldots\ldots\ldots, P$
$k = 1, 2, \ldots\ldots\ldots\ldots\ldots, K$

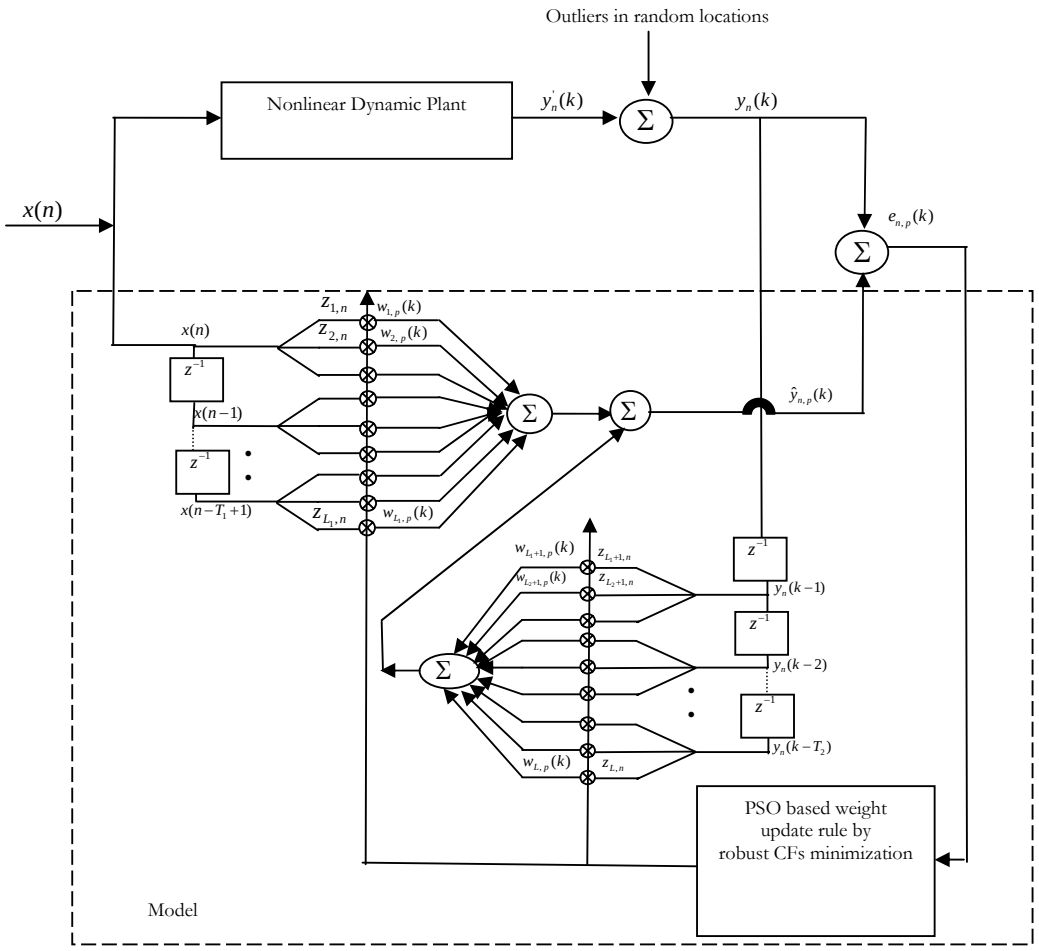


Fig. 6.2 Identification of nonlinear dynamic plants using FLANN architecture and PSO based robust CF minimization

6.3 Weight update of FLANN model by squared error minimization using PSO

The initial (k=0) position vector of each particle is represented by a single weight vector $\underline{W}_p(0)$ defined in (6.4) and is formed by combining the feed-forward and feedback weight vectors of the FLANN model. This initial set of vectors are obtained by generating random numbers lying between -0.5 to +0.5. The corresponding sets of initial velocity vector associated with the position vectors of particle is assumed to be random numbers lying between -0.5 to +0.5 and is denoted as $\Delta\underline{W}_p(0)$. For each particle, N input samples are applied successively and the corresponding initial squared error norm, $E_p(0)$ is computed using (6.3), (6.5) and (6.6). The initial best position vector of a particle corresponds to its own position vector. That is $\underline{W}_{b^p}(0) = \underline{W}_p(0)$ and the associated cost function of each particle is $E_p(0); 1 \le p \le P$. The initial potentiality of a p th particle is represented by the twin parameters $\{\underline{W}_{b^p}(0)$ and $E_p(0)\}$. The initial global best position $\underline{W}_g(0)$ is obtained by comparing all $\underline{W}_{b^p}(0)$ and choosing the one which provides minimum $E_p(0)$. In the next generation the velocity and position of each particle are updated as

$$\Delta\underline{W}_p(k+1) = \alpha\Delta\underline{W}_p(k) + c_1 * \underline{R}_1(k)[\underline{W}_{b^p}(k) - \underline{W}_p(k)] + c_2 * \underline{R}_2(k)[\underline{W}_g(k) - \underline{W}_{b_p}(k)] \quad (6.7)$$

$$\underline{W}_p(k+1) = \underline{W}_p(k) + \Delta\underline{W}_p(k+1) \quad (6.8)$$

where $\Delta\underline{W}_p(k)$ = velocity or rate of change of position change of p th particle vector at the k th generation of PSO.

$\underline{W}_p(k)$ = position vector of p th weight-particle at k th generation

$\underline{W}_{b^p}(k)$ = the best position vector of p th particle which yields the best fitness value until k th generation

$\underline{W}_g(k)$ = The best position vector among all the particles in the population achieved up to k th generation.

c_1 and c_2 represent positive constants. $\underline{R}_1$ and $\underline{R}_2$ represent two vectors of random numbers each of which lies in the range of 0 to 1. The second and third terms of (6.7) represent the self thinking of a particle and the social collaboration among particles respectively. α represents the inertia weight which balances between global and local

searches. It may be a fixed positive or a time varying constant. By linearly decreasing α from a relatively large value to a small value in succeeding generations, the particle tends to have a more global search ability in the beginning of the search but attains more local search ability towards the last generation [6.39].

To obtain the best position of a weight particle at k th generation the two successive fitness values $E_p(k-1)$ and $E_p(k)$ are compared and the weight vector associated with minimum fitness value is selected as its personal best position vector, $\underline{E}_{b_p}(k)$. This process is repeated for all particles. The global best weight vector, $\underline{W}_g(k)$ is then selected from among local best weight vectors which associate minimum fitness value. This process is repeated for many generations until the fitness function defined in (6.6) attains lowest possible minimum. The corresponding global best weight particle $\underline{W}_g(k)$ provides the desired solution. It means that this weight vector of the FLANN model generates output which is in close agreement with the plant output.

6.4 Development of robust identification and prediction models using PSO based training with robust norm minimization

Three robust cost functions reported in the literature are used in the development of identification model. The PSO is then used to iteratively minimize these norms of the error terms obtained from the model and hence the resulting identification model is expected to be robust. These cost functions are defined as follows

(a) Robust Cost Function-1 (Wilcoxon Norm) [6.1, 6.2]

A score function is first defined as an increasing function $\phi(u):[0,1]\rightarrow\Re$ such that

$$\int_0^1 \phi^2(u)du < \infty \tag{6.9}$$

The score function has the characteristics

$$\int_0^1 \phi(u)du = 0 \text{ and } \int_0^1 \phi^2(u)du = 1 \tag{6.10}$$

The score associated with the score function ϕ is defined as

$$a_\phi(i) = \phi\left(\frac{i}{l+1}\right), \quad i \in l \tag{6.11}$$

where l is a fixed positive integer.

From (6.10) it may be observed that $a_\phi(1) \le a_\phi(2) \le ..... \le a_\phi(l)$. The Wilcoxon norm [6.1, 6.2] is a pseudo-norm on $\Re^l$ and is defined as

$$C_1 = \sum_{i=1}^{l} a(R(v_i))v_i = \sum_{i=1}^{l} a(i)v_i, \quad v = [v_1, v_2, ..., v_l]^T \in \Re^l \tag{6.12}$$

where $R(v_i)$ denotes the rank of v_i among $v_1, v_2, ..., v_l$, $v_{(1)} \le v_{(2)} \le .....v_{(l)}$ are the ordered values of $v_1, v_2, ..., v_l$, $a(i) = \phi[i/(l+1)]$. In statistics, different types of score functions have been dealt but the commonly used one is given by $\phi(u) = \sqrt{12}(u - 0.5)$.

(b) Robust Cost Function-2 [6.47]

It is defined as

$$C_2 = \sigma(1 - \exp(-e^2/2\sigma)) \tag{6.13}$$

where σ = a parameter to be adjusted during training

and e^2 = mean square error defined in (6.6)

(c) Robust Cost Function – 3 (Mean Log Squared error) [6.44]

The third cost function is defined as

$$C_3 = \log(1 + \frac{e^2}{2}) \tag{6.14}$$

where e^2 is defined in (6.6).

The weight-update of the identification model of Fig. 6.2 is carried out by minimizing these cost functions of the errors defined in (6.12), (6.13) and (6.14) using PSO algorithm. In this approach the steps outlined from (6.3) to (6.5) remain same. Subsequent steps involved are detailed as follows :

Let the error vector of p th particle at k th generation due to application of N input samples to the model be represented as $[e_{1,p}(k), e_{2,p}(k), ..., e_{N,p}(k)]^T$. The errors are then arranged in an increasing manner from which the rank $R\{e_{n,p}(k)\}$ of each n th error term is obtained. The score associated with each rank of the error term is evaluated as

$$a(i) = \sqrt{12}\left(\frac{i}{N+1} - 0.5\right) \tag{6.15}$$

where $i,\ (1 \le i \le N)$ denotes the rank associated with each error term. At k th generation of each p th particle the Wilcoxon norm is then calculated as

$$C_p(k) = \sum_{i=1}^{N} a(i)\, e_{i,p}(k) \tag{6.16}$$

Similarly the other two CFs are computed using (6.13) and (6.14). The steps involved in the first and second generations in the PSO based minimization of these CFs are detailed in Figs. 6.3 and 6.4 respectively. The computations required in subsequent generations are just repetitions of the steps outlined in Fig. 6.4. The learning strategy described in Figs. 6.3 and 6.4 continues until the CF decreases to the possible minimum values. At this stage the training is complete and the global best weight vector $\underline{W}_g$ represents the feed forward and feedback weights of the FLANN based model.

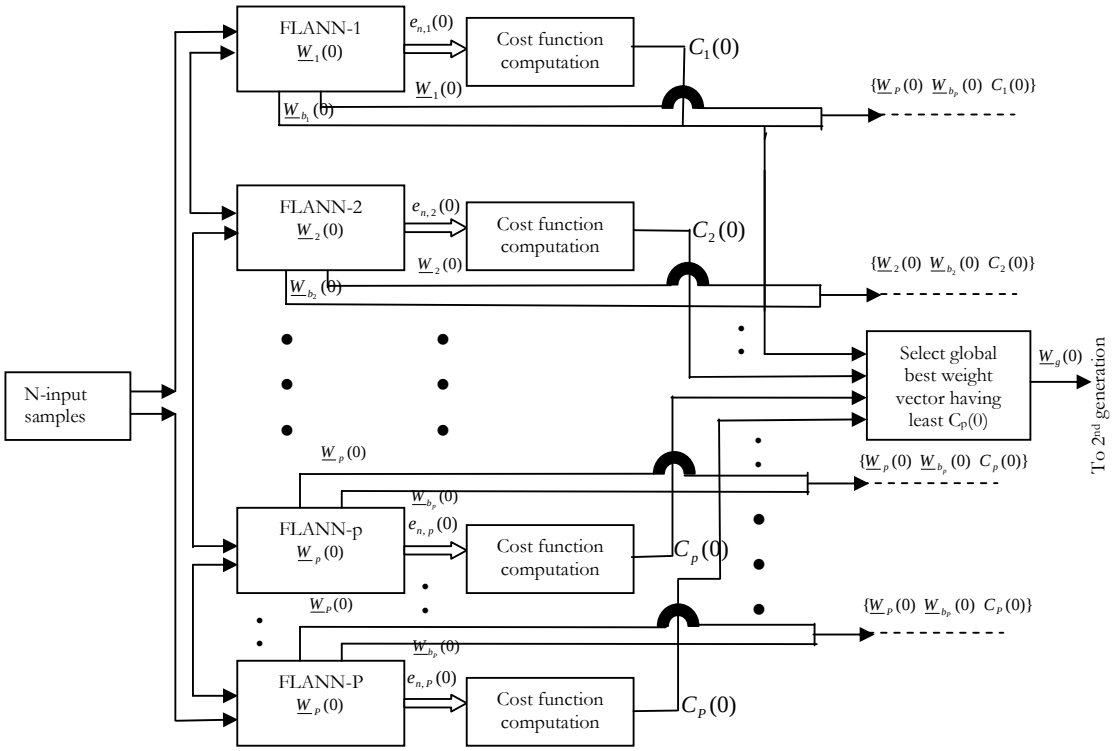


Fig. 6.3 Steps involved in the first generation weight update mechanism using PSO based CF minimization

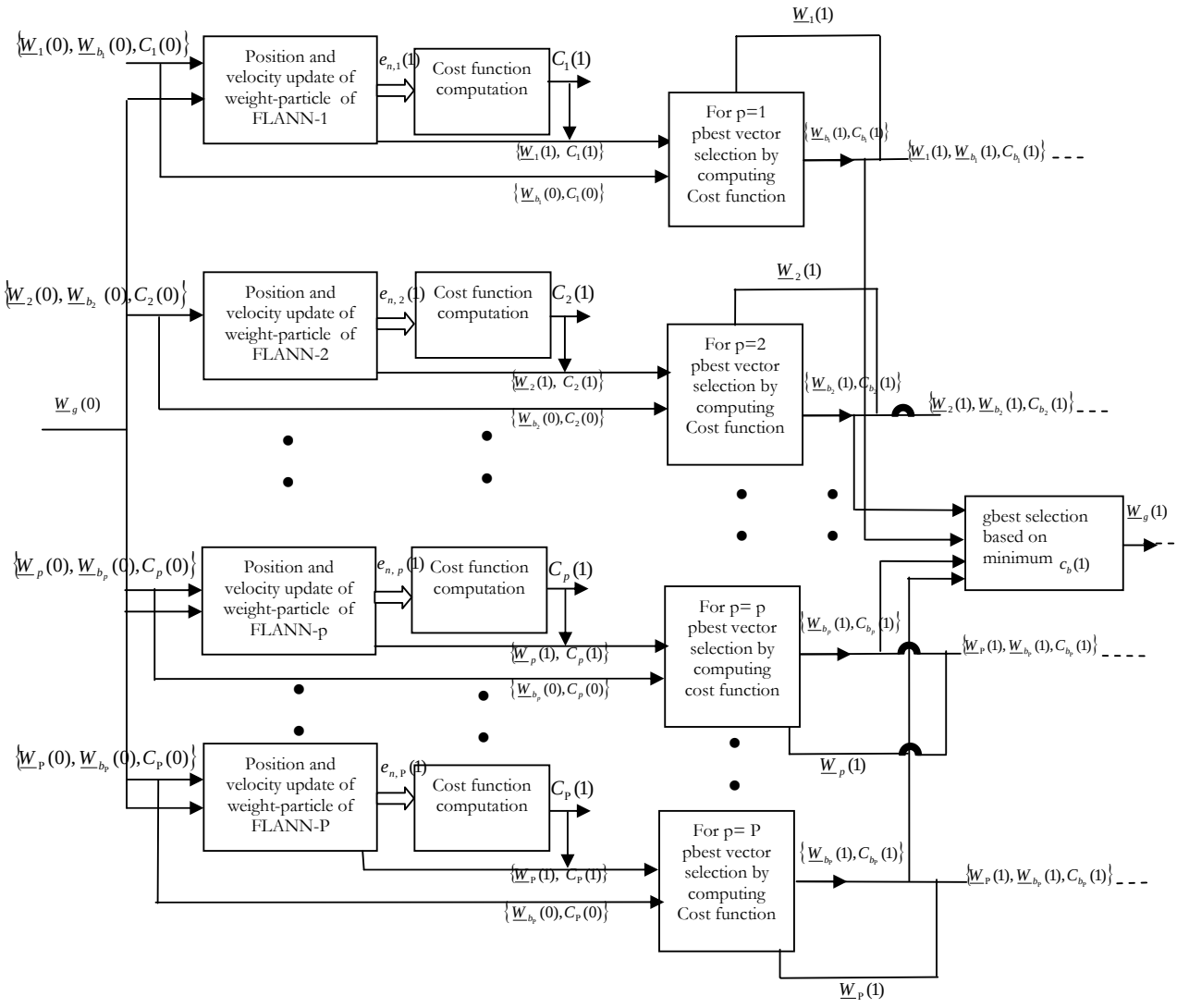


Fig. 6.4 Steps involved in 2nd generation weight-update mechanism using PSO based CF minimization

6.5 Simulation study

In this section, simulation study is carried out to assess the identification performance of the proposed algorithm and results of nonlinear identification of static and dynamic plants described by (3.3)-(3.6) are presented in presence of 10% to 50% of outliers in the desired signal. The outliers are uniformly distributed random values within the range of -1 to +1 and are added at random locations (10% to 50%) of the training samples. In these

examples, the series-parallel model is used to identify these plants. In the scheme-1, MSE is used as the cost function where as in schemes-2, 3 and 4 the CFs used are C_1, C_2 and C_3 respectively. The performance of the proposed three schemes are obtained from simulation studies and compared with that obtained by scheme-1. For training the weights of FLANN model an uniformly distributed random signal in the interval [-1,1] is used as input. During the testing phase, the effectiveness of the proposed models are evaluated by using the test signal

$$x(k) = \begin{cases} \sin\dfrac{2\pi k}{250} \ \ for\ k \le 250 \\ 0.8\sin\dfrac{2\pi k}{250} + 0.2\sin\dfrac{2\pi k}{25} \ \ for\ k > 250 \end{cases} \tag{6.17}$$

A quantitative measure for performance evaluation used is the normalized mean square error (NMSE) defined in [6.51] as

$$\Gamma = \frac{1}{\sigma^2 S}\sum_{k=1}^{S}[y(k) - \hat{y}(k)]^2 \tag{6.18}$$

where $y(k)$ and $\hat{y}(k)$ represent the plant and model outputs at k th discrete time, respectively and σ^2 denotes variance of the plant output sequence over S number of test samples.

Identification of Static Systems

Identification of two different static plants is carried out through simulation experiments.

Example 1 : $f_1(x) = x^3 + 0.3x^2 - 0.4x$ (6.19)

Example2: $f_2(x) = 0.6\sin(\pi x) + 0.3\sin(3\pi x) + 0.1\sin(5\pi x)$ (6.20)

Fig. 6.5 shows desired signal of Example-1 with outliers of 50% added to it within the range of (-1, 1). The input to the model is expanded into eleven trigonometric terms to get the best identification results in all schemes. The number of connecting weights including the threshold is twelve, which are updated using four different schemes. The parameters used in the study are no. of particles=30, no. of input samples=200, $c_1 = c_2 = 1.042$ *and* $v_{\max} = 1$. Simulation is carried out using 10% to 50% of outliers in the training samples but the results shown in Figs. 6.6(a) – (d) are for 50% outlier only. It is evident from these plots that the scheme-2 based model provides accurate response matching in presence of 50% of outlier whereas the scheme-1 based model exhibits poor identification performance. In both examples, NMSE obtained from scheme-2 listed in

Table 6.1 is much lower than that obtained from scheme-1 model.

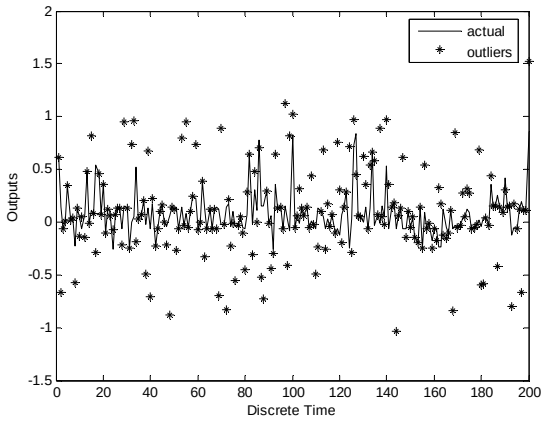
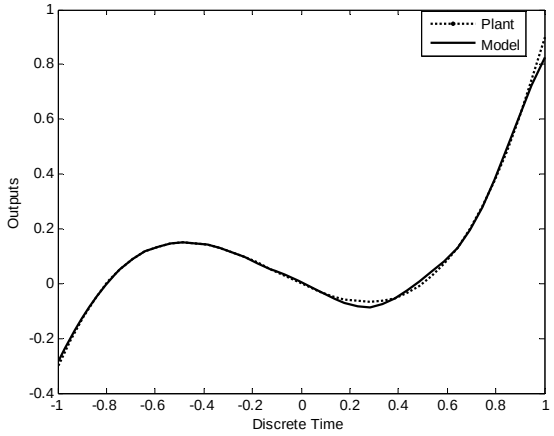
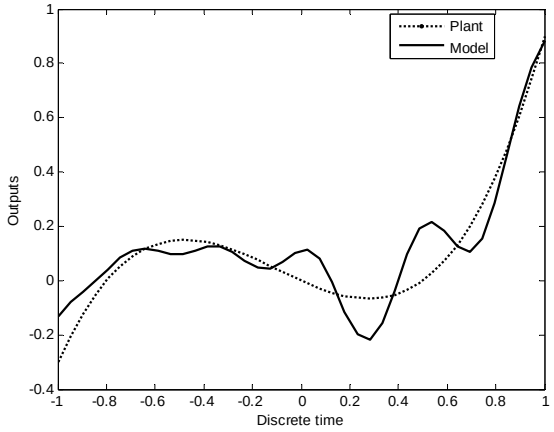


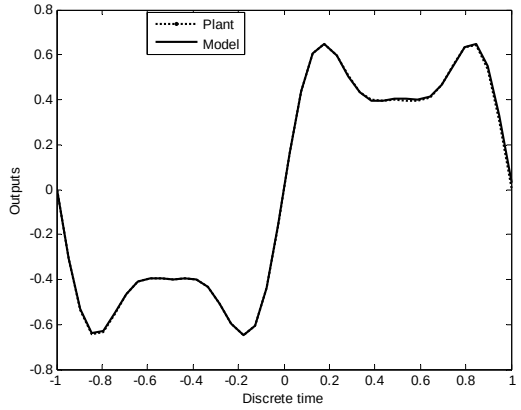
Fig. 6.5 Plot of desired signal with 50% outliers used in Example-1



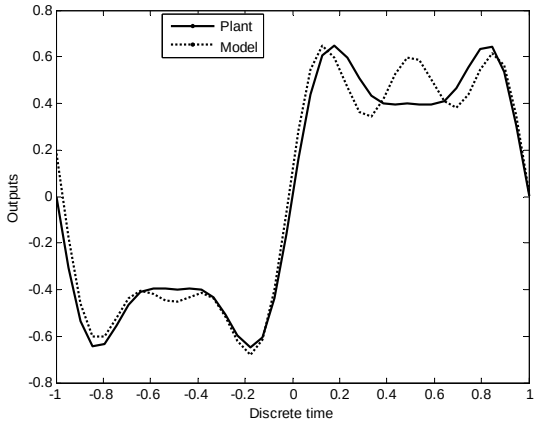
(a) Scheme-2 learning with 50% outliers



(b) Scheme-1 learning with 50% outliers



(c) Scheme-2 learning with 50% outliers



(d) Scheme-1learninmg with 50% outliers

Fig. 6.6 Response matching of static systems ((a) and (b) for Example 1 and (c) and (d) for Example 2)

Identification of SISO Dynamic Systems

Example 3: The difference equation of the plant is

$$y(k+1) = 0.3y(k) + 0.6y(k-1) + g[x(k)] \tag{6.21}$$

The linear parameters are 0.3 and 0.6 and two unknown nonlinearities $g_i(.)$ used in the study are

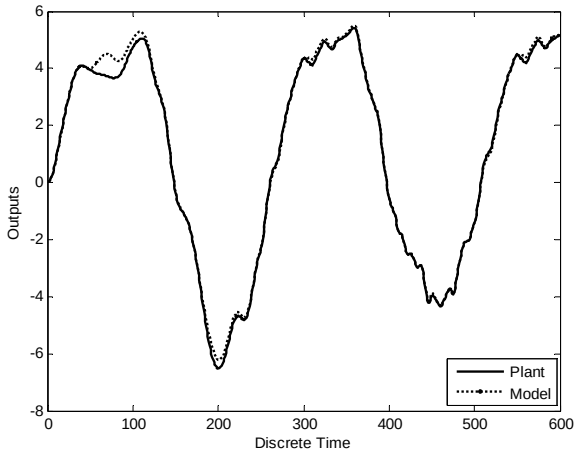
$$g_1(x) = 0.5\sin^3(\pi x) - \frac{2.0}{x^3 + 2.0} - 0.1\cos(4\pi x) + 1.125 \tag{6.22}$$

$$g_2(x) = 0.6\sin(\pi x) + 0.3\sin(3\pi x) + 0.1\sin(5\pi x) \tag{6.23}$$

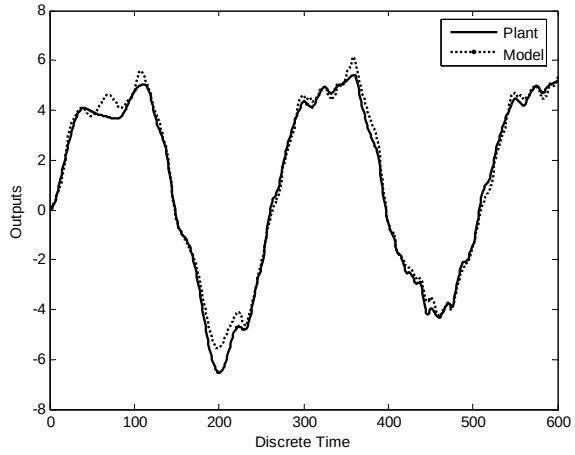
To identify the plant a series-parallel model described by (6.24) is used

$$\hat{y}(k+1) = 0.3y(k) + 0.6y(k-1) + \mathrm{N}[x(k)] \tag{6.24}$$

The term $\mathrm{N}[\mathrm{x}(\mathrm{k})]$ represents a FLANN model using various schemes. The input is expanded to 14 terms by using trigonometric expansions and PSO algorithm is used to update its connecting weights. The parameters used are no. of particles=30, no. of input samples=200, $c_1 = c_2 = 1.042$ *and* $v_{\max} = 1$. The results of identification of (6.21) with nonlinear functions defined in (6.22) and (6.23) in prersence of 50% and 40% outliers are shown in Figs. 6.7(a), (b) and Figs. 6.8(a), (b) respectively . It is observed that the FLANN with scheme-2 learning exhibits robust performance compared to that offered by the model using scheme-1. This is also supported by the NMSE listed in Table 6.1.

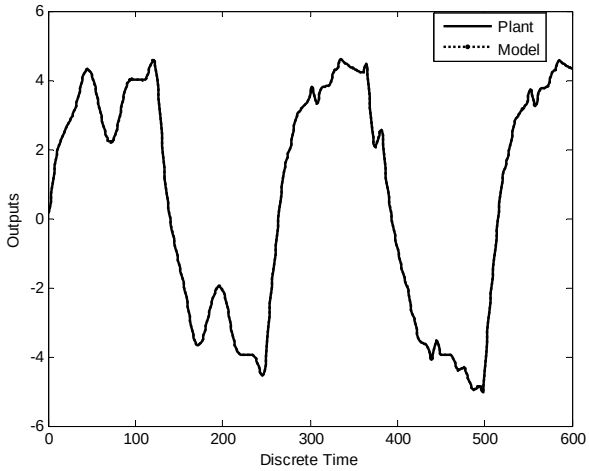


(a) Using Scheme-2 learning with 50% outliers

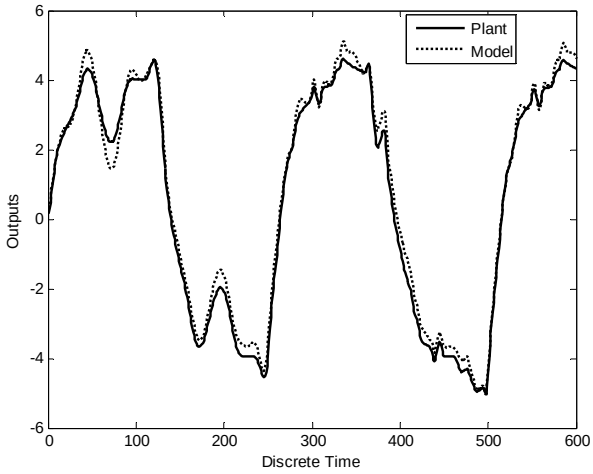


(b) Using Scheme -1 learning with 50% outliers

Fig.6.7 Comparison of response of the dynamic plant of Example 3 using nonlinearity defined in (6.22)



(a) Using Scheme-2 learning with 40% outliers



(b) Using Scheme-1 learning with 40% outliers

Fig. 6.8 Comparison of response of the dynamic plant of Example 3 using nonlinearity defined in (6.23)

Example 4 : In this example the plant to be identified is of Model-2 type and is represented by the difference equation

$$y(k+1) = f[y(k), y(k-1)] + x(k) \tag{6.25}$$

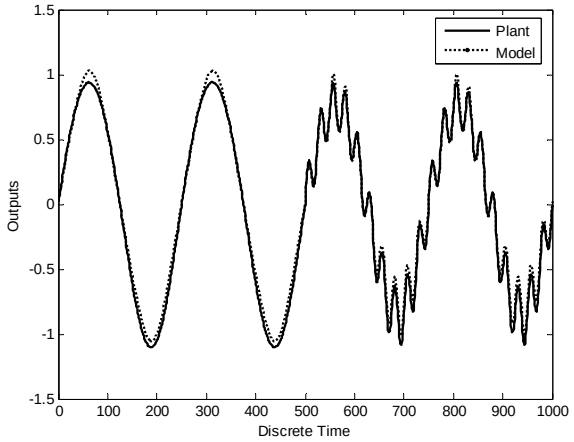
The unknown nonlinearity associated with the plant is given by

$$f(y_1, y_2) = \frac{y_1 y_2 (y_1 + 2.5)(y_1 - 1.0)}{1.0 + y_1^{\,2} + y_2^{\,2}} \tag{6.26}$$

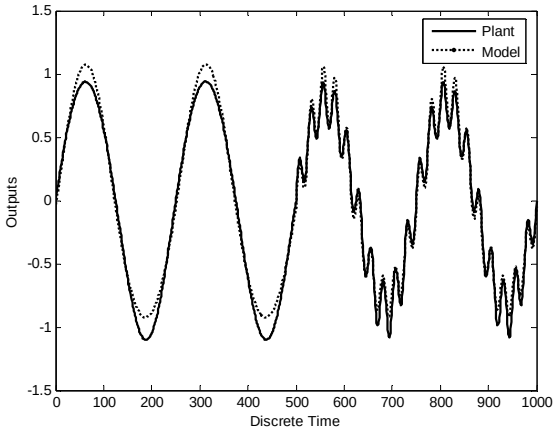
In this case the series-parallel scheme of the model is given by

$$\hat{y}(k+1) = \mathrm{N}[(y(k), y(k-1)] + x(k) \tag{6.27}$$

The two inputs are expanded into 12 terms and scheme-1 and scheme-2 based training are used. The parameters used are no. of particles=30, no. of input samples =500, $c_1 = c_2 = 1.042$ *and* $v_{\max} = 1$. The response obtained from the plant and the two models are shown in Figs. 6.9(a) and (b). These figures and Table 6.1 indicate that scheme-2 provides robust identification compared to that offered by scheme-1 method.



(a) Using Scheme-2 learning with 50% outliers



(b) Using Scheme-1 learning with 50% outliers

Fig. 6.9 Comparison of response of the dynamic plant of Example 4

Example 5: In this case the plant is of Model-3 type and is given by the difference equation

$$y(k+1) = f[y(k)] + g[x(k)] \tag{6.28}$$

where the unknown nonlinear functions $f(.)$ and $g(.)$ are represented as

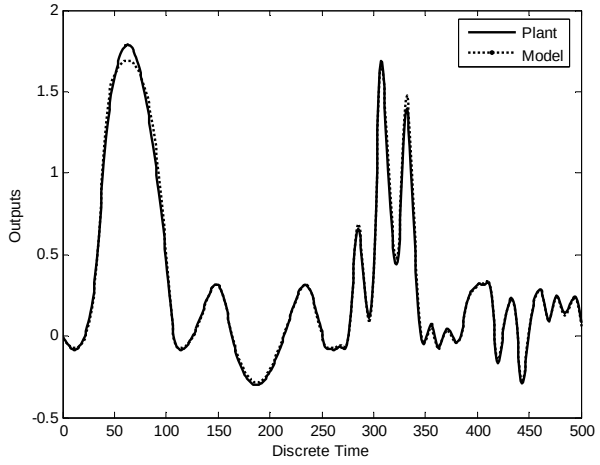
$$f(y) = \frac{y(y+0.3)}{1.0+y^2} \tag{6.29}$$

$$g(x) = x(x+0.8)(x-0.5) \tag{6.30}$$

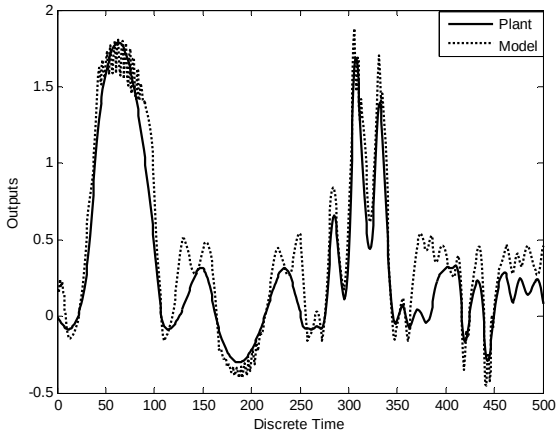
The model is represented by a series-parallel scheme

$$\hat{y}(k+1) = \mathrm{N}_1[y(k)] + \mathrm{N}_2[x(k)] \tag{6.31}$$

where N_1 and N_2 represent the FLANN model with scheme-1 or scheme-2 training. N_1 and N_2 structures contain seven and five number of expansions. For PSO the parameters used are no. of particles=30, no. of input samples =500, $c_1 = c_2 = 1.042$ *and* $v_{\max} = 1$. The response obtained from the plant and various models are compared in Figs. 6.10(a) and (b) and the computed NMSE is presented in Table 6.1. These results also indicate superior performance of the proposed scheme-2 technique over its scheme-1 counterpart.



(a) Using Scheme-2 learning with 50% outliers



(b)Using Scheme-1 learning with 50% outliers

Fig. 6.10 Comparison of response of the dynamic plant of Example 5

Example 6: The plant in this case is category Model-4 and is described by the difference equation

$$y(k+1) = f[y(k), y(k-1), y(k-2), x(k), x(k-1)] \tag{6.32}$$

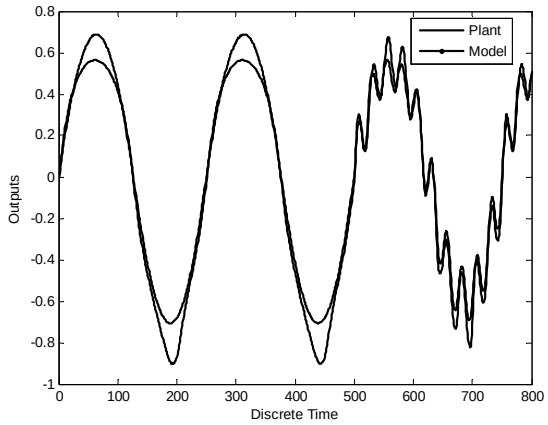
where the unknown nonlinear function f is given by

$$f[a_1, a_2, a_3, a_4, a_5] = \frac{a_1 a_2 a_3 a_5 (a_3 - 1.0) + a_4}{1.0 + a_2^{\,2} + a_3^{\,2}} \tag{6.33}$$

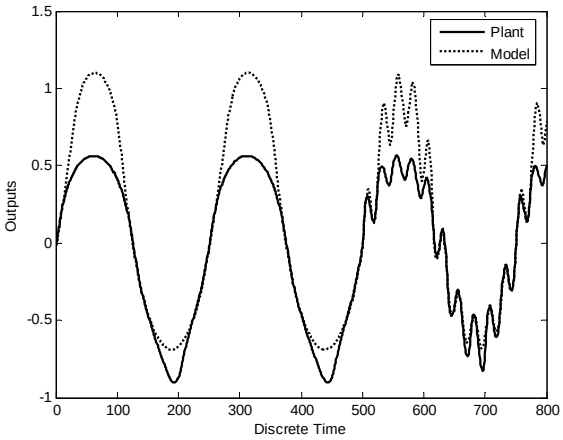
The series-parallel model used for identification of this plant is given as

$$\hat{y}(k+1) = \mathrm{N}[y(k), y(k-1), y(k-2), x(k), x(k-1)] \tag{6.34}$$

In case of scheme-1 and scheme-2 models, the input is expanded to six terms and output is expanded by nine terms. Figs. 6.11(a)-(b) show the comparative performance of the output response of two models. The simulation results also indicate that the identification performance is best in the proposed model as may be evident from comparison of NMSE shown in Table 6.1.



(a) Using Scheme-2 learning with 40% outliers



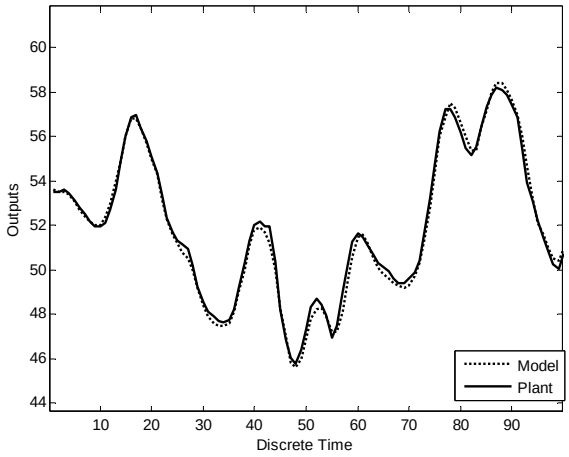
(b)Using Scheme-1 learning with 40% outliers

Fig. 6.11 Comparison of response of the dynamic plant of Example 6

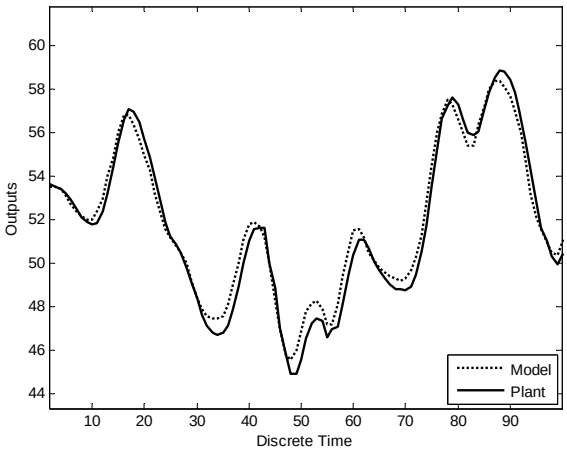
Example 7 : Identification of Box-Jenkin's System

A total of 296 pairs input-output samples are generated with a sampling period of 9s. The gas combustion process has one variable, gas flow $x(k)$ and one output variable, the concentration of CO_2, $y(k)$. The output $y(k)$ is influenced by four past output samples $y(k-1), y(k-2), y(k-3)$ and $x(k-1)$. Uniformly distributed random values between $[-3, 3]$ is added at 10% to 50% random location of the desired samples. Figs. 6.12 (a) and (b) display the actual and estimated values obtained by using scheme-2 and scheme-1

methods of training respectively. It is evident from these figures that scheme-2 provides better identification performance in comparison to scheme-1 based method in presence of strong outliers in the training signal.



(a) Using Scheme-2 learning with 50% outliers



(b) Using Scheme-1 learning with 50% Outliers

Fig. 6.12 Output response matching of Example 8

Example 8 : Prediction of Mackey Glass Time Series

The Mackey-Glass System (MGS) is a standard benchmark system for identification. This is a chaotic time series generated by solving the time-delay differential equation

$$\frac{dx(t)}{dt} = -bx(t) + a\frac{x(t-\tau)}{1+x(t-\tau)^{10}} \tag{6.35}$$

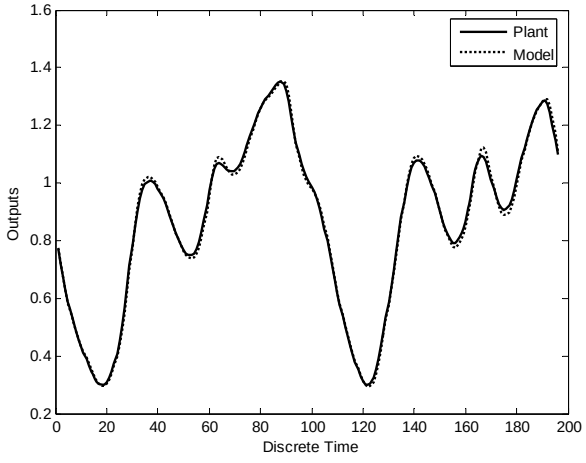
The MG Series is periodic for $\tau < 17$ and is non-periodic otherwise. Initial values are taken as random values. The differential equation is solved using Euler's method. A set of 1100 samples are generated with b = 0.9, a = 0.2 and $\tau = 30$. The first 100 samples are discarded due to their random nature. Out of the remaining 1000 samples, 800 samples are used as training data and the remaining 200 as test samples.

The model of the system can be represented by

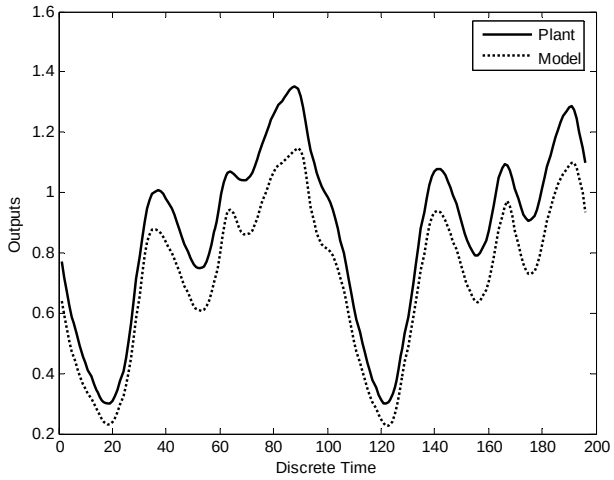
$$x(t+p) = f\{x(t), x(t-\tau), x(t-2\tau), ..., x(t-(N-1)\tau\} \quad (6.36)$$

where $p = 4$ and $N = 4$.

$x(t), x(t-\tau), x(t-2\tau), x(t-3\tau)$ are used as the inputs and $x(t+p)$ is used as the output. The training data set is corrupted by adding random values from a uniform distribution of [−15, 15] to the uncorrupted data set. Simulation is carried out in presence of 10% to 50% of outliers in the training signal. The result of response matching is shown in Figs. 6.13(a) and (b) for 50% outlier only. From these figures it is observed that scheme-2 based model identify the system correctly in presence of 50% outlier in the training signal where as the scheme-1 based model fails to identify the system.



(a) Using Scheme-2 learning with 50% outliers

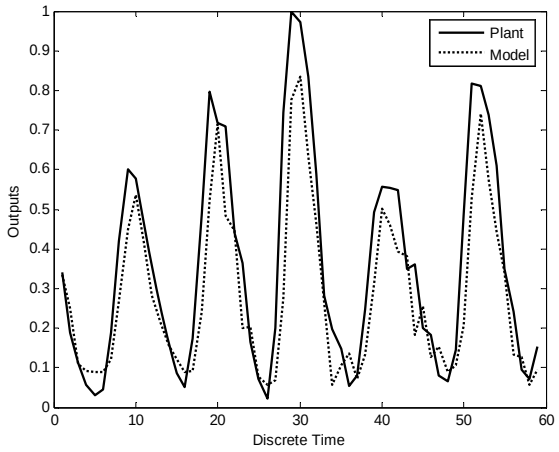


(b) Using Scheme-1learning with 50% outliers

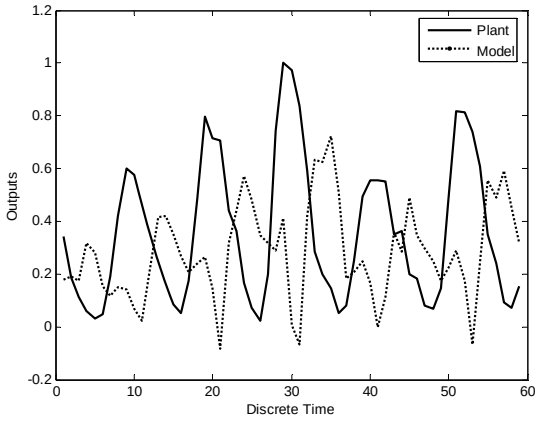
Fig. 6.13 Output response matching of Example 8

Example 9 : **Prediction of Sunspot Time Series**

The series consists of 288 data points of yearly averages of sunspots starting from the year 1700 to the year 1987. The sunspots problem is a typical time series prediction problem, in which the sunspots number is to be predicted for the following year based on data of the past years. Out of the 288 data, first 225 data are used for training and rest 63 data used for testing purpose. The training data set is corrupted by adding random values from a uniform distribution defined between [−15, 15] to the uncorrupted data. Simulation is carried out in presence of 10% to 50% of outliers in the training signal. The response matching of the system with 40% outliers and NMSE obtained for 10% to 40% outliers is given in Figs. 6.14(a) and (b). The identification performance of scheme-1 is severely degraded at 40% outliers. It is clearly observed that the schem-2 model performs better in all cases in comparison to the scheme-1 based model in presence of outliers.



(a) Using Scheme-2 learning with 40% outliers



(b) Using Scheme-1 learning with 40% outliers

Fig. 6.14 Output response matching of Example 9

Table 6.1

Comparison of NMSE obtained in Example-1 to Example-6 from models using three robust cost functions and conventional MSE CF

Example No.	% of outliers	NMSE (in dB)			
		RCF-1	RCF-2	RCF -3	MSE
1	10	-19.85,	-17.10	-14.05	-12.58
	20	-28.86,	-18.23	-10.91	-9.64
	30	-26.43,	-16.74	-11.63	-10.14
	40	-27.42,	-15.51	-10.56	-9.33
	50	-24.03	-16.69	-11.59	-8.67
2	10	-34.52	-22.98	-19.38	-19.06
	20	-42.46	-23.49	-16.91	-15.88
	30	-36.64	-21.14	-17.89	-16.52
	40	-35.78	-23.80	-16.82	-15.77
	50	-37.10	-22.29	-15.70	-14.91
3 with (6.22)	10	-26.44	-21.88	-21.78	-20.80
	20	-32.87	-22.65	-21.50	-20.39
	30	-34.82	-21.62	-20.51	-19.70
	40	-30.56	-21.99	-20.98	-19.87
	50	-27.79	-20.83	-20.19	-19.23
3 with (6.23)	10	-38.51	-27.98	-19.92	-19.15
	20	-42.70	-27.80	-20.41	-19.27
	30	-38.45	-26.23	-20.51	-19.58
	40	-52.52	-25.84	-20.93	-20.55
	50	-37.92	-25.97	-20.63	-20.19
4	10	-23.82	-22.87	-22.00	-21.43
	20	-24.97	-24.14	-23.04	-20.48
	30	-28.55	-24.84	-24.09	-21.80
	40	-26.21	-21.49	-21.04	-19.71
	50	-22.56	-18.55	-18.01	-18.31
5	10	-14.73	-9.36	-8.37	-4.07
	20	-14.49	-13.77	-11.45	-9.74
	30	-15.65	-14.76	-10.58	-8.30
	40	-17.39	-14.97	-9.12	-4.87
	50	-23.82	-8.23	-5.11	-4.53
6	10	-16.37	-16.86	-16.53	-16.53
	20	-16.36	-11.91	-7.35	-5.07
	30	-13.18	-12.42	-9.74	-8.71
	40	-13.80	-13.26	-7.91	-5.53
	50	-13.53	-12.23	-11.26	-11.28

6.6 Conclusion

In this Chapter a novel method of robust identification of nonlinear dynamic system using a low complexity single layer FLANN model has been proposed. The robust identification task is formulated as an optimization of RCFs of the error terms of the model. The connecting weights of the FLANN model are iteratively adjusted using PSO technique to achieve this objective. The proposed technique is robust because it provides excellent identification performance of complex plants even when the training signal of the model contains up to 50% of outliers. The robust performance of the model using different RCFs is demonstrated using simulation of wide varieties of benchmark examples. The introduction of the new CFs in the model and PSO based minimization of these CFs are contributing to robust and improved performance compared to standard squared error norm based model. Further comparison of identification performance between the models using different RCFs indicate that the second scheme which employs Wilcoxon norm as the CF outperforms other three schemes. Robust identification performance using Wilcoxon norm is also observed in case of Mackey Glass, Box Jenkin's and Sunspot time series when strong outliers are also present in training set.

References

[6.1] Joseph W. McKean, "Robust analysis of Linear models", Statistical Science, vol. 19, no. 4, pp. 562-570, 2004.

[6.2] Jer-Guang Hsieh, Yih-Lon Lin and Jyh-Horng Jeng, "Preliminary study on Wilcoxon learning machines", IEEE Trans. on neural networks, vol. 19, no. 2, pp. 201-211, Feb. 2008.

[6.3] S. Haykin , Neural Networks, Ottawa, ON Canada: Maxwell Macmillan, 1994.

[6.4] P. S. Sastry, G. Santharam and K. P. Unnikrishnan, "Memory neural networks for identification and control of dynamical systems", IEEE Trans. Neural Networks, vol. 5, pp. 306-319, 1994.

[6.5] A. G. Parlos, K. T. Chong and A. F. Atiya, "Application of recurrent multilayer perceptron in modeling of complex process dynamics", IEEE Trans. Neural Networks, vol. 5, pp. 255-266, 1994.

[6.6] K. S Narendra. and K. Parthasarathy, "Identification and control of dynamical systems using neural networks". IEEE Trans. on neural networks, vol. 1, no. 1, pp. 4-27, 1990.

[6.7] D. H. Nguyen and B. Widrow, "Neural networks for self-learning control system". Int. J. Conttr., vol. 54, no. 6, pp. 1439-1451, 1991.

[6.8] G. Cembrano, G. Wells, J. Sarda and A. Ruggeri, "Dynamic control of a robot arm based on neural networks", Contr. Eng. Practice, vol 5, no. 4, pp. 485-492, 1997.

[6.9] S. Chen, S. A. Billings and P. M. Grant, "Recursive hybrid algorithm for nonlinear system identification using radial basis function networks", Int. J. Contr., vol. 55, no. 5, pp. 1051-1070, 1992.

[6.10] S. V. T. Elanayar and Y. C. Shin, "Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems", IEEE Trans. Neural Network, vol. 5, pp. 594-603, 1994.

[6.11] Q. Zhang and A. Benveniste, "Wavelet networks". IEEE Trans. Neural Networks, vol. 3, pp. 889-898, 1992.

[6.12] Y. C. Pati and P. S. Krishnaprasad, "Analysis and synthesis of feed forward neural networks using discrete affine wavelet transforms", IEEE Trans. Neural Networks, vol. 4, pp. 73-85, 1993.

[6.13] J. C. Patra, R. N. Pal, B. N. Chatterji and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks", IEEE Trans. in Systems, Man and Cybernetics-Part B, vol. 29, no. 2, pp. 254-262, 1999.

[6.14] J. C. Patra and A. C. Kot, "Nonlinear dynamic system identification using Chebyshev functional link artificial neural networks", IEEE Trans. in Systems, Man and Cybernetics-Part B, vol. 32, no. 4, pp. 505-511, 2002.

[6.15] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," IEEE Trans. Power Syst., vol. 15, no. 4, pp. 1232–1239, Nov. 2000.

[6.16] B. Zhao, C. Guo, and Y. J. Cao, "A multiagent-based particle swarm optimization approach for optimal reactive power dispatch," IEEE Trans. Power Syst., pp. 1070–1078, May 2005.

[6.17] J. Park, K. Lee, J. Shin, and K. Y. Lee, "A particle swarm optimization for economic dispatch with nonsmooth cost functions," IEEE Trans.Power Syst., vol. 20, no. 1, pp. 34–42, Feb. 2005.

[6.18] T. Aruldoss, A. Victoire, and A. Jeyakumar, "Reserve constrained dynamic dispatch of units with valve-point effects," IEEE Trans. Power Syst., vol. 20, no. 3, pp. 1273–1282, Aug. 2005.

[6.19] Z. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," IEEE Trans. Power Syst., vol. 18, no. 3, pp. 1187–1195, Aug. 2003.

[6.20] Y-P Chen, W. C. Peng and M. C. Jian, "Particle swarm with recombination and dynamic linkage discovery", IEEE Trans. on System, Man and Cybernetics – Part B, vol. 37, no. 6, pp. 1460-1470, Dec. 2007.

[6.21] Kassabalidis, M. El-Sharkawi, R. Marks, L. Moulin, and A. Silva, "Dynamic security border identification using enhanced particle swarm optimization," IEEE Trans. Power Syst., pp. 723–729, Aug.2002.

[6.22] S. Kannan, S. Slochanal, and N. Padhy, "Application of particle swarm optimization technique and its variants to generation expansion problem," ELSERVIER Electric Power Syst. Res., vol. 70, no. 3, pp.203–210, Aug. 2004.

[6.23] S. Kannan, S. Slochanal, and N. Padhy, "Application and comparison of metaheuristic techniques to generation expansion planning problem," IEEE Trans. Power Syst., vol. 20, no. 1, pp. 466–475, Feb. 2005.

[6.24] A. Abido, "Optimal power flow using particle swarm optimization," Int. J. Elect. Power Energy Syst., vol. 24, no. 7, pp. 563–571, Oct. 2002.

[6.25] J. Vlachogiannis and K. Lee, "Determining generator contributions to transmission system using parallel vector evaluated particle swarm optimization," IEEE Trans. Power Syst., vol. 20, no. 4, pp. 1765–1774, Nov. 2005.

[6.26] M. Abido, "Optimal design of power-system stabilizers using particle swarm optimization," IEEE Trans. Energy Conversion, vol. 17, no. 3, pp. 406–413, Sep. 2002.

[6.27] Z. Gaing, "A particle swarm optimization approach for optimum design of PID controller in AVR system," IEEE Trans. Energy Conversion, vol. 19, no. 2, pp. 384–391, Jun. 2004.

[6.28] J. Chia-Feng, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," IEEE Trans. Syst., Man, Cybern.,Part B: Cybern., vol. 34, no. 2, pp. 997–1006, Apr. 2004.

[6.29] Y. Liu, X. Zhu, J. Zhang, and S.Wang, "Application of particle swarm optimization algorithm for weighted fuzzy rule-based system," in Proc. 30th Annu. Conf. IEEE Ind. Electron. Soc., Nov. 2004, vol. 3, pp. 2188–2191.

[6.30] A. Esmin, G. Torres, and A. Zambroni, "A hybrid particle swarm optimization applied to loss power minimization," IEEE Trans. PowerSyst., vol. 20, no. 2, pp. 859–866, May 2005.

[6.31] X.Yu, X. Xiong, and Y.Wu, "A PSO-based approach to optimal capacitor placement with harmonic distortion consideration," Electric Power Syst. Res., vol. 71, pp. 27–33, Sep. 2004.

[6.32] C. Huang, C. J. Huang, and M. Wang, "A particle swarm optimization to identifying the ARMAXmodel for short-term load forecasting," IEEE Trans. Power Syst., vol. 20, no. 2, pp. 1126–1133, May 2005.

[6.33] J. Vlachogiannis and K. Lee, "Determining generator contributions to transmission system using parallel vector evaluated particle swarm optimization," IEEE Trans. Power Syst., vol. 20, no. 4, pp. 1765–1774, Nov. 2005.

[6.34] S. H. Ling, H. H. C. Iu, K. Y. Chan, H. K. Lam, Benny C. W. Yeung and Frank H. Leung, "Hybrid particle swarm optimization with wavelet mutation and its industrial applications", IEEE Trans. on System, Man and Cybernetics – Part B, vol. 38, no. 3, pp. 743-763, June 2008.

[6.35] Shinn-Ying Ho, Hung-Sui Lin, Weei-Hurng Liauh and Shinn-Jang Ho, "OPSO : Orthogonal particle swarm optimization and its application to task assignment problems", IEEE Trans. on Systems, Man and Cybernetics – Part A, vol. 38, no. 2, pp. 288-298, March 2008.

[6.36] Alberto Moraglio, Cecilia Di Chio, Julian Togelius and Riccardo Poli, "Geometric particle swarm optimization" Journal of Artificial Evolution and Applications, ID 143624, pp. 1-14, 2008.

[6.37] L. S. Coelho, "Novel Gaussian quantum behaved particle swarm optimizer applied to electromagnetic design", IET Sci. Meas. Technol., vol. 1, no. 5, pp. 290-294, 2007.

[6.38] T. O. Ting, M. V. C Rao and C. K. Loo, "A novel approach for unit commitment problem via an effective hybrid particle swarm optimization", IEEE Trans. on Power Systems, vol. 21, no. 1, pp. 411-418, Feb. 2006.

[6.39] J. J. Liang, A. K. Qin, Ponnuthurai Nagaratnam Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions", IEEE Trans. on Evolutionary Computation, vol. 10, no. 3, pp. 281-295, June 2006

[6.40] David S. Chen and Ramesh C. Jain, "A robust back propagation learning algorithm for function approximation", IEEE Trans. on Neural Networks, vol. 5, no. 3, pp. 467-479, May 1994.

[6.41] Jerome T. Connor, R. Douglas Martin and L. E. Atlas, "Recurrent neural networks and robust time series prediction", IEEE Trans. on Neural Networks, vol. 5, no. 2, pp. 240-254, March 1994.

[6.42] V. David Sanchez A., "Robustization of a learning method for RBF networks", Neurocomputing, vol. 9, pp. 85-94, 1995.

[6.43] Kadir Liano, "Robust error measure for supervised neural network learning with outliers", IEEE Trans. On Neural Networks, vol. 7, no. 1, pp. 246-250, Jan. 1996.

[6.44] Wei-Yen Wang, Tsu-Tian Lee, Ching-Lang Liu and Chi-Hsu Wang, "Function approximation using fuzzy neural networks with robust learning algorithm", IEEE Trans. on Systems, Man and Cybernetics-Part B : Cybernetics, vol. 27, no. 4, pp. 740-747, Aug. 1997.

[6.45] Lei Huang, Bai-Ling Zhang and Qian Huang, "Robust interval regression analysis using neural networks", Fuzzy Sets and Systems, vol. 97, pp. 337-347, 1998.

[6.46] Chien-Cheng Lee, Pau-Choo Chung, Jea-Rong Tsai and Chein-I Chang, "Robust radial basis function neural networks", IEEE Trans. on Systems, Man and Cybernetics – Part B : Cybernetics, vol. 29, no. 6, pp. 674-685 , Dec., 1999.

[6.47] Hung-Hsu Tsai and Pao-Ta Yu, "On the optimal design of fuzzy neural networks with robust learning for function approximation", IEEE Trans. n Systems, Man and Cybernetics-Part B : Cybernetics, vol. 30, no. 1, pp. 217-223, Feb. 2000.

[6.48] Chen-Chia Chuang, Shun-Feng Su and Song-Shyong Chen, "Robust TSK fuzzy modeling for function approximation with outliers", IEEE Trans. on Fuzzy Systems, vol. 9, no. 6, pp. 810-821, Dec. 2001

[6.49] Chen-Chia Chuang, Shun-Feng Su, Jin-Tsong Jeng and Chih-Ching Hsiao, "Robust support vector regression networks for function approximation with outliers", IEEE Trans. on Neural networks, vol. 13, no. 6, pp. 1322-1330, Nov. 2002.

[6.50] Y. H. Pao., Adaptive Pattern Recognition & Neural Networks, Reading, MA : Addison-Wesley, 1989.

[6.51] N. A. Gershenfeld and A. S. Weigend, "The future of time series: Learning and understanding", in Time Series Prediction: Forecasting the future and past, Reading, MA: Addison-Wesley, pp. 1-70, 1993.

Chapter

7

Robust Adaptive Inverse Modeling using Bacterial Foraging Optimization Technique and Applications

7.1 Introduction

THE inverse model of a system having an unknown transfer function is itself a system having a transfer function which is in some sense a best fit to the reciprocal of the unknown transfer function. Sometimes the inverse model response contains a delay which is deliberately incorporated to improve the quality of the fit. In Fig. 7.1, a source signal $s(n)$ is fed into an unknown system that produces the input signal $x(n)$ for the adaptive filter. The output of the adaptive filter is subtracted from a desired response signal that is a delayed version of the source signal, such that

$$d(n) = s(n-\Delta) \tag{7.1}$$

where Δ is a positive integer value. The goal of the adaptive filter is to adjust its characteristics such that the output signal is an accurate representation of the delayed source signal.

There are many applications of adaptive inverse model of a system. If the system is a communication channel then the inverse model is an adaptive equalizer which compensates the effects of inter symbol interference (ISI) caused due to restriction of channel bandwidth [7.1]. Similarly if this system is the model of a high density recording medium then its corresponding inverse model reconstruct the recorded data without distortion [7.5]. If the system represents a nonlinear sensor then its inverse model represents a compensator of environmental as well as inherent nonlinearities [7.44]. The adaptive inverse model also finds applications in adaptive control [7.4] as well as in deconvolution in geophysics application [7.3].

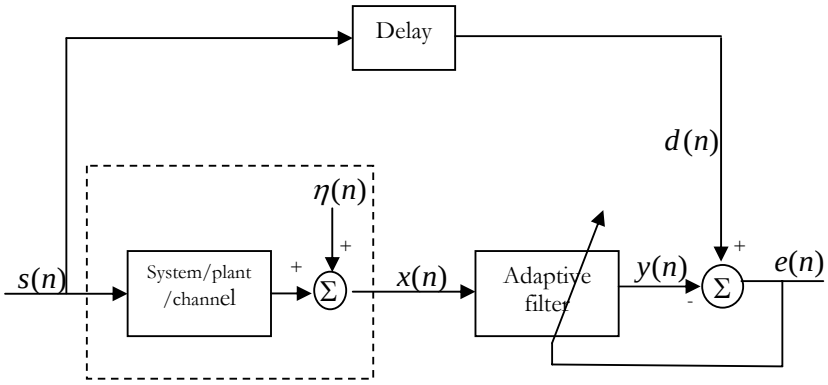


Fig. 7.1 Inverse Modeling

Channel equalization is a technique of decoding of transmitted signals across nonideal communication channels. The transmitter sends a sequence $s(n)$ that is known to both the transmitter and receiver. However, in equalization, the received signal is used as the input signal $x(n)$ to an adaptive filter, which adjusts its characteristics so that its output closely matches a delayed version $s(n-\Delta)$ of the known transmitted signal. After a suitable

adaptation period, the coefficients of the system either are fixed and used to decode future transmitted messages or are adapted using a crude estimate of the desired response signal that is computed from $y(n)$. This latter mode of operation is known as decision-directed adaptation.

Channel equalization is one of the first applications of adaptive filters and is described in the pioneering work of Lucky [7.1]. Today, it remains as one of the most popular uses of an adaptive filter. Practically every computer telephone modem transmitting at rates of 9600 bits per second or greater contains an adaptive equalizer. Adaptive equalization is also useful for wireless communication systems. Qureshi [7.2] has written an excellent tutorial on adaptive equalization. A related problem to equalization is deconvolution, a problem that appears in the context of geophysical exploration [7.3].

In many control tasks, the frequency and phase characteristics of the plant hamper the convergence behavior and stability of the control system. We can use an adaptive filter shown in Fig. 7.1 to compensate for the nonideal characteristics of the plant and as a method for adaptive control. In this case, the signal *s(n)* is sent at the output of the controller, and the signal *x(n)* is the signal measured at the output of the plant. The coefficients of the adaptive filter are then adjusted so that the cascade of the plant and adaptive filter can be nearly represented by the pure delay $z^{-\Delta}$. Details of the adaptive algorithms as applied to control tasks in this fashion can be found in [7.4].

Transmission and storing of high density digital information plays an important role in the present age of information technology. Digital information obtained from audio, video or text sources needs high density storage or transmission through communication channels. Communication channels and recording medium are often modeled as band-limited channel for which the channel impulse response is that of an ideal low pass filter. When a sequence of symbols are transmitted/recorded, the low pass filtering of the channel distorts the transmitted symbols over successive time intervals causing symbols to spread and overlap with adjacent symbols. This resulting linear distortion is known as inter symbol interference. In addition nonlinear distortion is also caused by cross talk in the channel and use of amplifiers. In the data storage channel, the binary data is stored in the form of tiny magnetized regions called bit cells, arranged along the recording track. At read back, noise and nonlinear distortions (ISI) corrupt the signal. An ANN based equalization technique has been proposed [7.5] to alleviate the ISI present during read back from the magnetic

storage channel. Recently, Sun et al have reported [7.6] an improved Viterbi detector to compensate the nonlinearities and media noise. Thus adaptive channel equalizers play an important role in recovering digital information from digital communication channels/storage media. Preparta had suggested [7.7] a simple and attractive scheme for dispersal recovery of digital information based on the discrete Fourier transform. Subsequently Gibson et al have reported [7.8] an efficient nonlinear ANN structure for reconstructing digital signal which has passed through a dispersive channel and corrupted with additive noise. In a recent publication [7.9] the authors have proposed an optimal preprocessing strategies for perfect reconstruction of binary signals from a dispersive communication channels. Touri et al have developed [7.10] deterministic worst case frame work for perfect reconstruction of discrete data transmission through a dispersive communication channel. In recent past, new adaptive equalizers have been suggested using soft computing tools such as artificial neural network (ANN), polynomial perceptron network (PPN) and the functional link artificial neural network (FLANN) [7.11]. It is reported that these methods are best suited for nonlinear and complex channels. Recently, Chebyshev artificial neural network has also been proposed for nonlinear channel equalization[7.12]. The drawback of these methods are that the estimated weights may likely fall to local minima during training.

For this reason genetic algorithm (GA) has been suggested for training adaptive channel equalizers[7.13]. The main attraction of GA lies in the fact that it does not rely on Newton-like gradient-descent methods, and hence there is no need for calculation of derivatives. This makes them less likely to be trapped in local minima. But only two parameters of GA, the crossover and the mutation, help to avoid local minima problem. There is still some situations when the weights in GA optimization are trapped to local minima.

In recent years bacterial foraging optimization (BFO) has been proposed [7.14] and has been applied in harmonic estimation of power system signals[7.15], adaptive inverse modeling[7.16], image segmentation[7.17], image filtering[7.18], optimal power flow[7.19], economic load dispatch[7.20 -7.21], parameter estimation[7.22], independent component analysis[7.23], recognition of handwriting[7.24], tuning of power system stabilizers[7.25], controller optimization[7.26], design of multiple optimal power system stabilizers[7.27], optimization of coefficients of PI controller[7.28-7.30], optimization in dynamic environments[7.31], Hammerstein model identification[7.32], D-STATCOM[7.33], function minimization and control[7.34], load compensation [7.35], load forecasting[7.36]. A hybrid GA and BFO algorithm

has been developed for global optimization in [7.37]. Adaptation of run length unit by using Takagi-Sugeno fuzzy scheme based on the minimum value of cost function has been reported in [7.15] .The chemotactic step size is made adaptive to accelerate the convergence speed near the optima[7.38] and mathematical analysis of reproduction operator is reported in [7.39]. The BFO is an useful alternative to GA and requires less number of computations. In addition, the BFO is also a derivative free optimization technique. The number of parameters that are used for searching the total solution space is higher in BFO compared to those in GA. Hence the possibility of avoiding the local minimum is higher in BFO. In this scheme, the foraging (methods for locating, handling and ingesting food) behaviour of E. Coli bacteria present in our intestines is mimicked.

In case of the derivative free algorithms conventionally the mean square error (MSE) is used as the fitness or cost function. Use of MSE as cost function leads to improper training of adaptive model when outliers are present in the desired signal. It is a fact that the traditional regressors employ least square fit which minimizes the Euclidean norm, while the robust estimator is based on a fit which minimizes another rank based on a norm called Wilcoxon norm [7.40]. It is known in statistics that linear regressors developed using Wilcoxon norm are robust against outliers. Using such norm new robust machines have recently been proposed for approximation of nonlinear functions [7.41]. In the present investigation we develop a new method of robust inverse model of complex nonlinear channels and systems by minimizing robust cost function (RCF) [7.41], [7.42] and [7.43] of errors of the model using a derivative free BFO technique. The performance of the new method is evaluated through simulation study and is compared with the results obtained from corresponding error square norm based BFO technique.

7.2 Data recovery by adaptive channel equalization

Reading out of high density data from the recording medium or recovery of binary data from the noisy digital channel needs ISI compensation. This is achieved by employing an adaptive inverse model shown in Fig. 7.2. The transmitted symbols are represented as $x(k)$ at time instance, k. They are then passed into the channel model which may be linear or nonlinear. An FIR filter is used to model a linear channel whose output at time instant k may be written as

$$y(k) = \sum_{i=0}^{N-1} w(i)x(k-i) \tag{7.2}$$

where $w(i)$ are the channel tap values and N is the length of the FIR system or channel. The "NL" block represents the nonlinear distortion of the symbols in the channel and its output may be expressed as

$$\begin{aligned} z(k) &= \psi(x(k), x(k-1), \ldots\ldots x(k-N+1) \\ &\quad w(0), w(1), \ldots\ldots \ldots\ldots \ldots\ldots \ldots\ldots w(N-1)), \end{aligned} \tag{7.3}$$

where $\psi(.)$ is some nonlinear function generated by the "NL" block. The channel output $z(k)$ is corrupted with additive white Gaussian noise $q(k)$ of variance σ^2. This corrupted received signal is given by $r(k)$. The received signal $r(k)$ is then passed into the digital channel equalizer to produce $\hat{x}(k)$ which recovers the transmitted symbol $x(k)$. From initial tap values (at $t = 0, w(i) = 0$), the weights are updated until the cost function, $\sum_{k=1}^{N} e^2(k)$, is minimized. Where $N =$ No. of input samples used for training and $e(k) = x_d(k) - \hat{x}(k)$. The received or recorded data is given by $r(k) = z(k) + q(k)$. The minimization of this cost function is iteratively performed by BFO scheme which is dealt in the Chapter 2

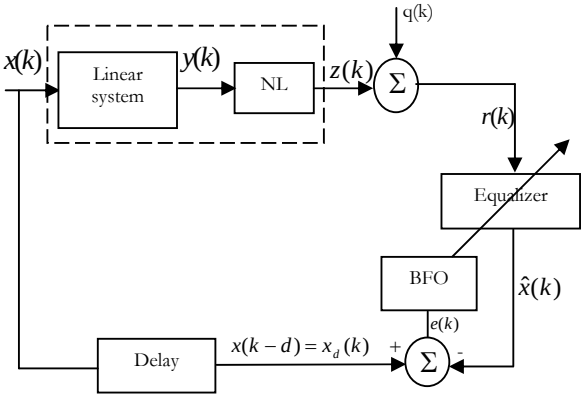


Fig. 7.2 A Digital Communication System with BFO based adaptive inverse model

7.3 BFO based training of weights of inverse model

The updating of the weights of the BFO based inverse model is carried out using the training rule as outlined in the following steps:

Step -1 **Initialization of various parameters**

(i) S_b = No. of bacteria to be used for searching the total region

(ii) N_{is} = Number of input samples

(iii) p = Number of parameters to be optimized

(iv) N_s = Swimming length after which tumbling of bacteria will be undertaken in a chemotactic loop.

(v) N_c = Number of iterations to be undertaken in a chemotactic loop. Always $N_c > N_s$.

(vi) N_{re} = Maximum number of reproduction to be undertaken

(vii) N_{ed} = Maximum number of elimination and dispersal events to be imposed over the bacteria.

(viii) P_{ed} = Probability with which the elimination and dispersal continue.

(ix) The location of each bacterium P (1- p , 1- S_b , 1) is specified by random numbers on [0,1].

(x) The value of $C(i)$ (i.e. run length unit). It is assumed to be constant for all bacteria.

Step-2 **Generate desired signal**

(i) Random binary input [1,-1] is applied to the channel..

(ii) The output of the channel is contaminated with white Guassian noise of known strength to produce the input signal for the equalizer.

(iii) The binary input is delayed by half of the order of the equalizer to act as the desired signal, $d(k)$.

Step -3 **Iterative algorithm for optimization**

This section models the bacterial population, chemotaxis, reproduction, elimination and dispersal. Initially $j = k = l = 0$

(i) Elimination dispersal loop $l = l + 1$

(ii) Reproduction loop $k = k + 1$

(iii) Chemotaxis loop $j = j + 1$

(a) For $i = 1,2,.....S_b$, the cost function, (in this case mean squared error) $J(i, j, k, l)$ for each i th bacterium is calculated as follows :

(1) N_{is} number of binary input are passed through the equalizer.

(2)The output is then compared with the corresponding desired signal, $d(k)$ to calculate the error, $e(k)$.

(3)The sum of squared error averaged over N_{is} is finally stored in $J(i, j, k, l)$.

(4)End of For Loop.

(b)For $i = 1,2,....., S_b$ the tumbling/swimming decision is taken.

Tumble : Generate a random vector $\Delta(i)$, with each element, $\Delta_m(i)$, $m = 1,2,.....p$, a random number in the range of [-1, 1].

Move: Let $P^i(j+1,k,l) = P^i(j,k,l) + C(i) \times \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$

This results in an adaptable step size in the direction of tumble for bacterium i. The cost function (mean squared error) $J(i, j+1, k, l)$ is computed.

Swim – (i) Let c =0; (counter for swim length)

(ii) While $c < N_s$ (have not climbed down too long)

Let $c = c + 1$

If $J(j) < J(j-1)$ then $P^i(j+1,k,l) = P^i(j,k,l) + C(i) \times \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$

and the $P(j+1,k,l)$ is used to compute the new $J(i, j+1, k, l)$

ELSE let $c = N_s$. This is the end of the WHILE statement.

(c)Go to next bacterium $(i+1)$ if $i \neq S_b$ to process the next bacterium.

(d)If $\min(J)$ {minimum value of J among all the bacteria} is less than the tolerance limit then break all the loops.

Step-4. If $j < N_c$, go to (iii) i.e. continue chemotaxis loop since the life of the bacteria is not over.

Step-5 **Reproduction**

(a) For the given k and l, and for each $i = 1,2,....., S_b$ let J^i be the health of the i th bacterium. The bacteria are sorted in ascending order of cost J (higher cost means lower health).

(b) The $S_r = \frac{S_b}{2}$ bacteria with highest J value die and other S_r bacteria with the best value split and the copies that are made are placed at the same location as their parent.

Step-6. If $k < N_{re}$ go to Step-2. In this case, the number of specified reproduction steps has not reached and the next generation in the chemotactic loop is to be started.

Step-7. **Elimination –Dispersal**

The bacterium, which has an elimination-dispersal probability above a preset value P_{ed}, is eliminated by dispersing to a random location and new replacements are randomly initialized over the search space. By this the total population is maintained constant.

7.4 Development of robust inverse modeling using BFO based training with robust norm minimization

Three robust cost functions defined in literature [7.41-7.43] are used in the development of robust adaptive inverse models. The BFO algorithm is then used to iteratively minimize these norms of the error obtained from the model and hence the resulting inverse model is expected to be robust against outliers. These cost functions are defined in Section 6.4 of Chapter 6. The weight-update of inverse model of Fig. 7.2 is carried out by minimizing these cost functions of the errors defined in (6.16), (6.17) and (6.18) using BFO algorithm. In this approach, the procedure outlined from Step-1 to Step-7 of section 7.3 remains the same. The only exception is detailed as follows :

Let the error vector of p th bacterium at k th generation due to application of N input samples to the model be represented as $[e_{1,p}(k), e_{2,p}(k),.....,e_{N,p}(k)]^T$. The errors are then arranged in an increasing manner from which the rank $R\{e_{n,p}(k)\}$ of each n th error term is obtained. The score associated with each rank of the error term is evaluated as

$$a(i) = \sqrt{12}\left(\frac{i}{N+1} - 0.5\right) \tag{7.4}$$

where $(1 \le i \le N)$ denotes the rank associated with each error term. At k th generation of each p th particle the Wilcoxon norm is then calculated as

$$C_p(k) = \sum_{i=1}^{N} a(i)\, e_{i,p}(k) \tag{7.5}$$

Similarly other two CFs are computed using (6.10), (6.17) and (6.18). The learning process continues until the CF decreases to the possible minimum values. At this stage the training is completed and the resulting weight vector represents the final weights of the inverse model.

7.5 Simulation study

In this section, the simulation study of the proposed inverse model in presence of 10% to 50% of outliers in the desired signal is carried out. Fig. 7.2 is simulated for various nonlinear sensor/systems/channels using the algorithm given in section7.4. Three standard linear systems used in the simulation study are :

$$\begin{aligned} &S1: 0.209 + 0.995z^{-1} + 0.209z^{-2} \\ &S2: 0.260 + 0.930z^{-1} + 0.260z^{-2} \\ &S3: 0.304 + 0.903z^{-1} + 0.304z^{-2} \end{aligned} \tag{7.6}$$

The eigen value ratio (EVR) of S1, S2 and S3 are 6.08, 11.12 and 21.71 respectively [7.11] . This ratio indicates the severity or contouring capacity of the system or channel. To study the effect of nonlinearity on the equalization performance, two different nonlinearities are introduced to the channel

$$\begin{aligned} &NL1: z(k) = \tanh(y(k)) \\ &NL2: z(k) = y(k) + 0.2y^2(k) - 0.1y^3(k) \end{aligned} \tag{7.7}$$

where $y(k)$ is the output of each of these linear systems (S1 through S3). The additive noise in the channel or measurement noise in sensor is white Gaussian with -30dB strength. In this study an 8-tap adaptive FIR filter is used as an inverse model. The desired signal is generated by delaying the input binary sequence by half of the order (4 in this case) of the inverse model. Outliers are added by simply replacing the bit value from 1 to -1 or -1 to 1 at randomly selected locations (10% to 50%) of the desired signal. In this simulation work, we have considered the following parameters of BFO : S_b = 8, N_{is} = 100, p =8, N_s =3, N_c =5, N_{re} =40-60, N_{ed} =10, P_{ed} = 0.25, $C(i)$ = 0.0075. For the sake of clarity different cost functions of the inverse models used in the simulation are mentioned here. These are

CF1 - Wilcoxon norm

CF2 - mean square error, e^2

CF3 - $\sigma(1-\exp^{\frac{-e^2}{2\sigma}})$, where σ is a constant.

CF4 - $\log\left(1+\frac{e^2}{2}\right)$

The bit error ratio (BER) plot of BFO trained inverse model pertaining to different nonlinear channels/sensors with different cost functions with 0%-50% of outliers are obtained through simulation and are plotted in the Figs. 7.3(a)-(f) to 7.8(a)-(f). The BER was computed for the nonlinear channels/systems with different EVR values at SNR 15dB in presence of 0% and 50% outliers in the desired signal and the results are plotted in Figs. 7.9 and 7.10 for 40% and 50% outliers respectively. Few notable observations obtained from these plots are :

(a)Keeping CF, SNR and percentage of outliers in the desired signal same, the BER increases with increase in the EVR of the channel. Similarly under identical conditions of simulation the squared error cost function based model performs the worst where as the Wilcoxon norm based model provides the best performance (least BER).

(b)As the outliers in the desired signal increases the Wilcoxon norm based model continues to provide lowest BER performance compared to that provided by other norms.

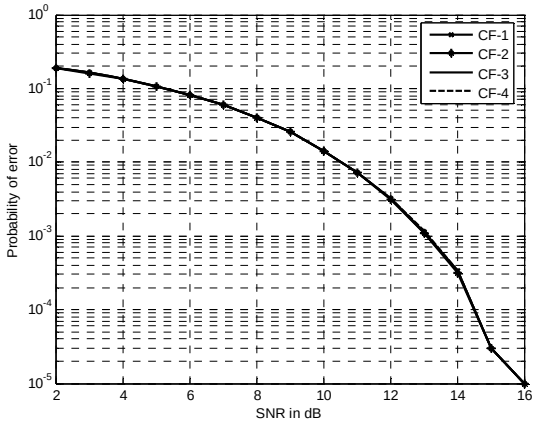
(c)With no outlier in the desired signal, the BER plot of all four CFs are almost same (Figs. 7.3(a), 7.4(a), 7.5 (a), 7.6 (a), 7.7(a) and 7.8(a)).

(d)At high outliers the conventional CF2 based model performs the worst followed by CF4 based model. In all cases the Wilcoxon norm (CF1) based inverse model performs the best and hence is more robust against low to high outliers in the training signal.

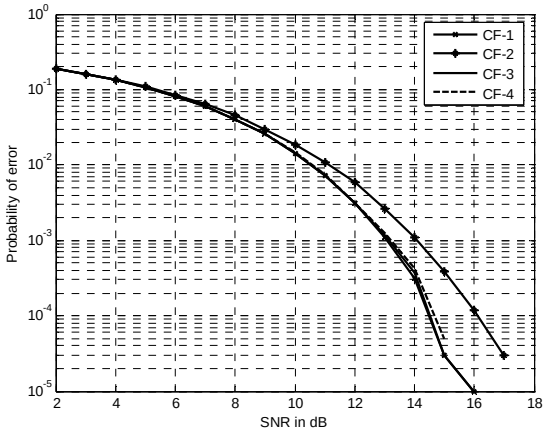
(e)The accuracy of inverse models based on CF3 and Cf4 norms developed using outliers is almost identical.

(f)In addition, the plots of Figs. 7.9 and 7.10 indicate that at 50% outliers in the desired signal the BER increases with increase in the EVR of the nonlinear channels or systems.

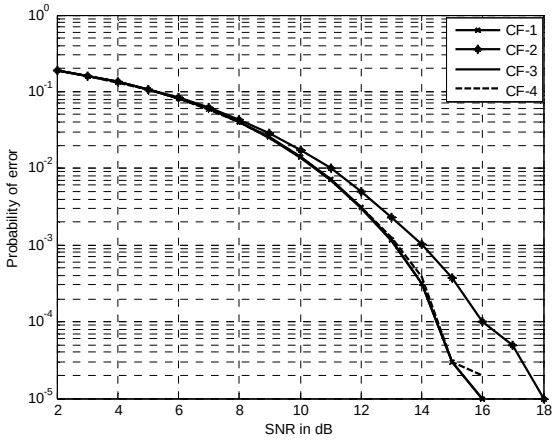
(g) Further, the BER of the inverse model in all channels and SNR conditions is highest in square error norm (CF2) based training compared to the all other three norms used. However the Wilcoxon norm (CF1) based inverse model yields minimum BER among all cases studied.



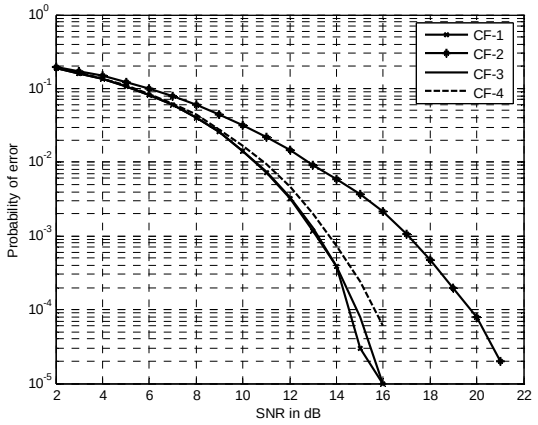
(a) 0% Outliers



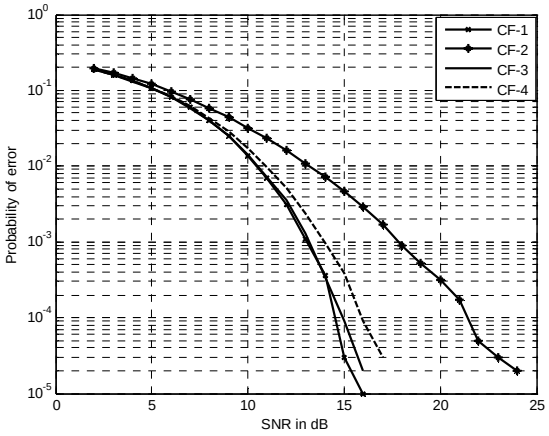
(b)10% Outliers



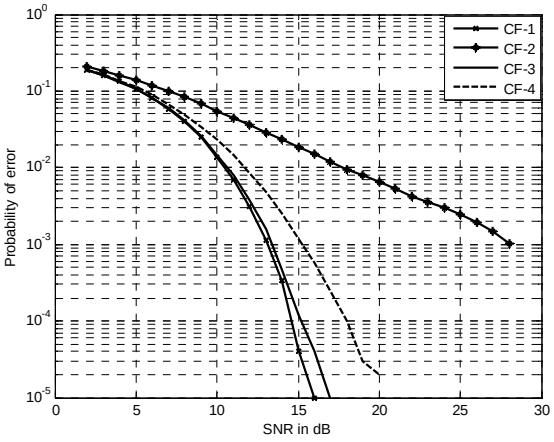
(c) 20% Outliers



(d) 30% Outliers

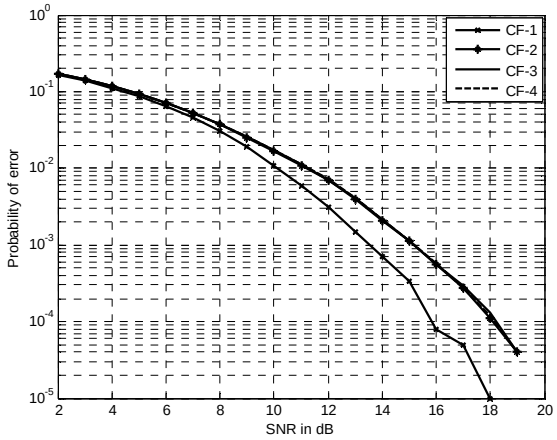


(e) 40% Outliers

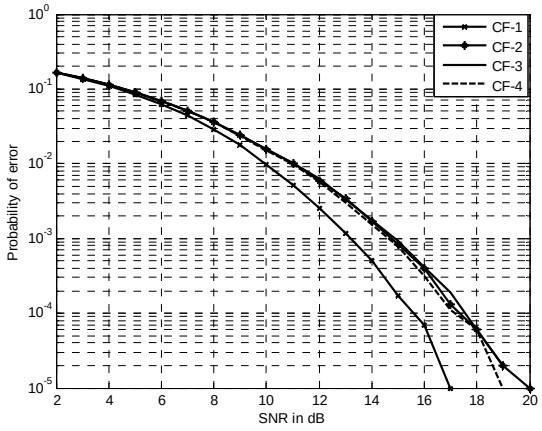


(f) 50% Outliers

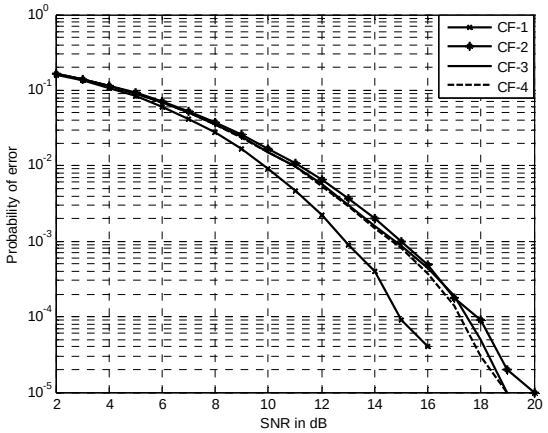
Fig. 7. 3. Comparison of BER of four different CFs based nonlinear equalizers with [.209, .995, .209] as channel coefficients and NL1



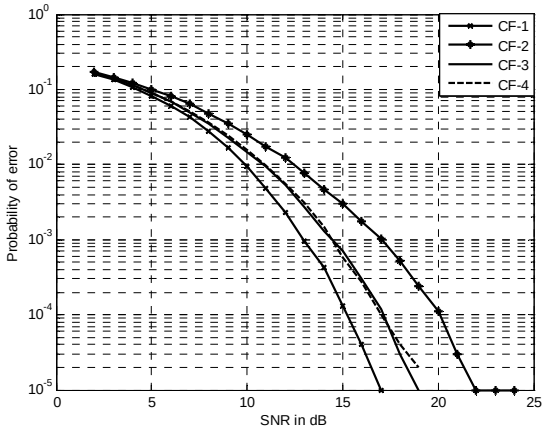
(a) 0% Outliers



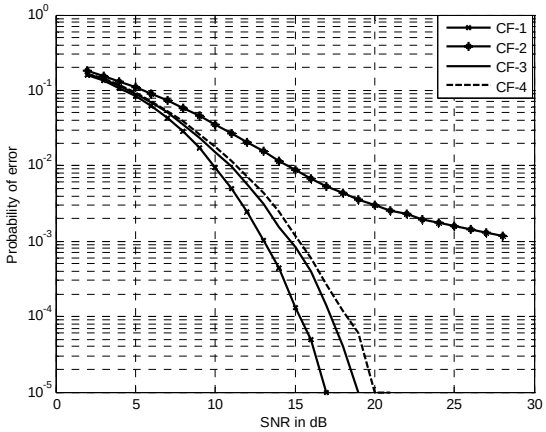
(b) 10% Outliers



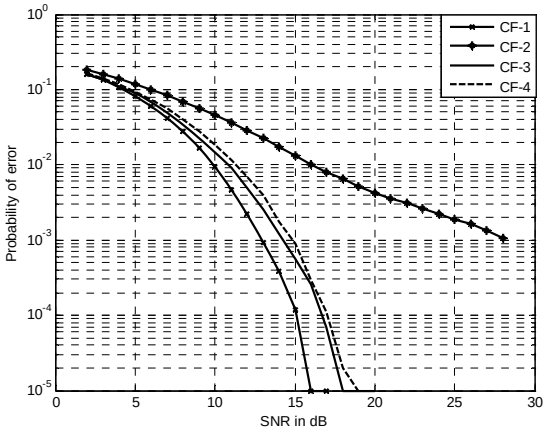
(c) 20% Outliers



(d) 30% Outliers

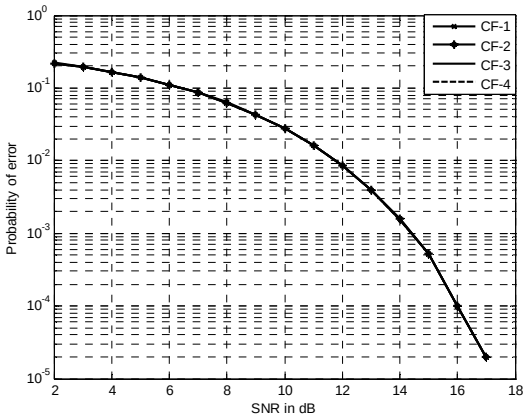


(e) 40% Outliers

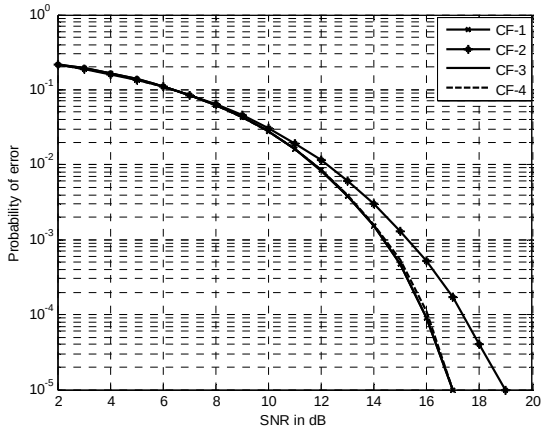


(f) 50% Outliers

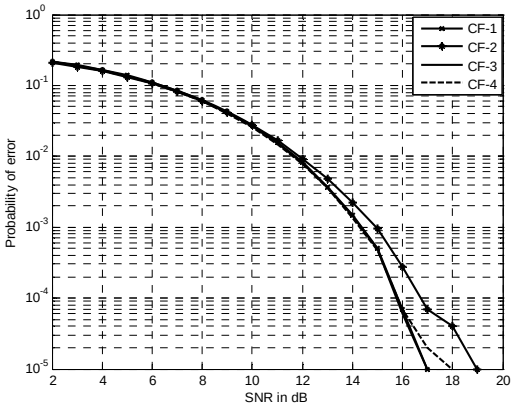
Fig. 7. 4. Comparison of BER of four different CFs based nonlinear equalizers with [.209, .995, .209] as channel coefficients and NL2



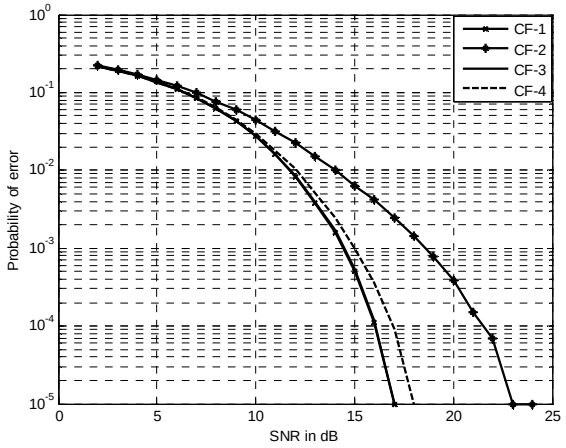
(a) 0% Outliers



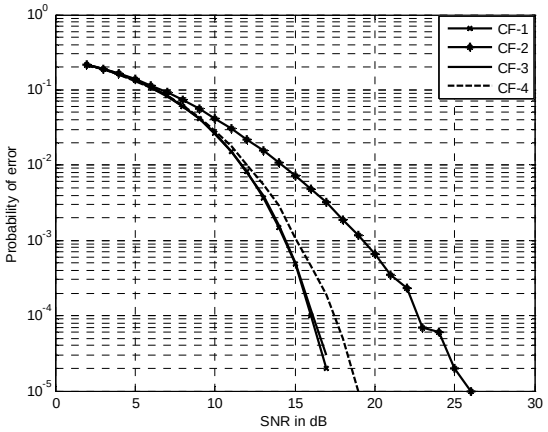
(b) 10% Outliers



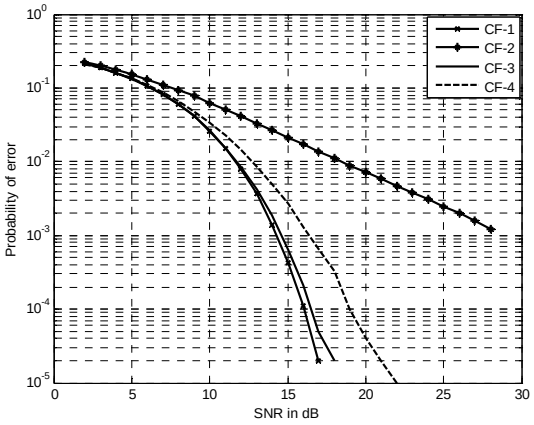
(b) 20% Outliers



(d) 30% Outliers

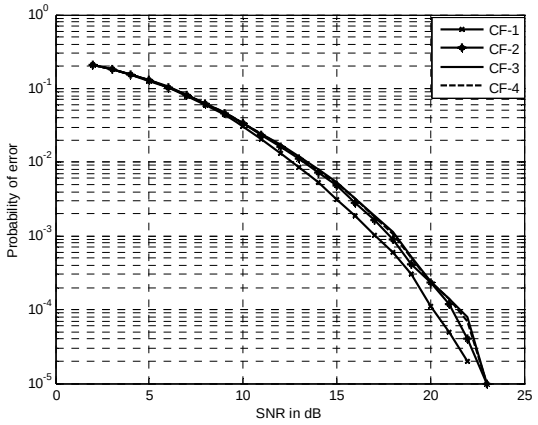


(e) 40% Outliers

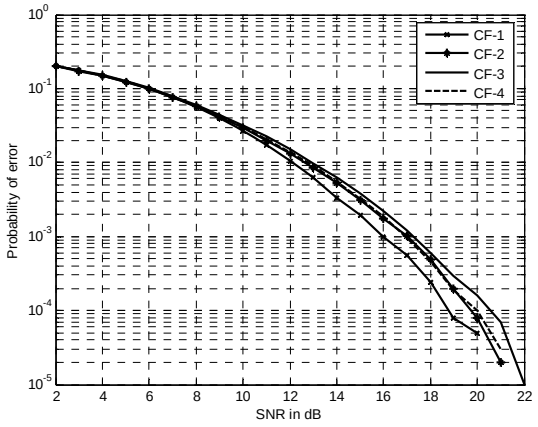


(f) 50% Outliers

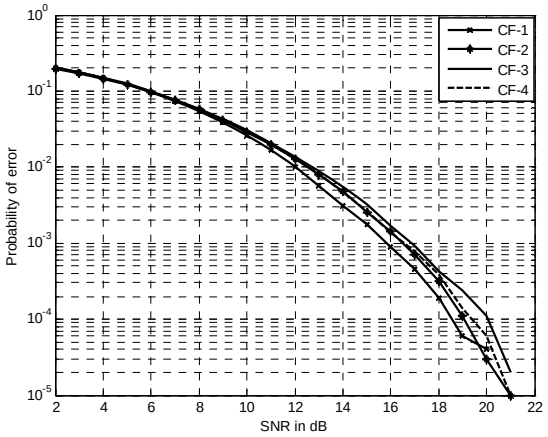
Fig. 7.5. Comparison of BER of four different CFs based nonlinear equalizers with [.260, .930, .260] as channel coefficients and NL1



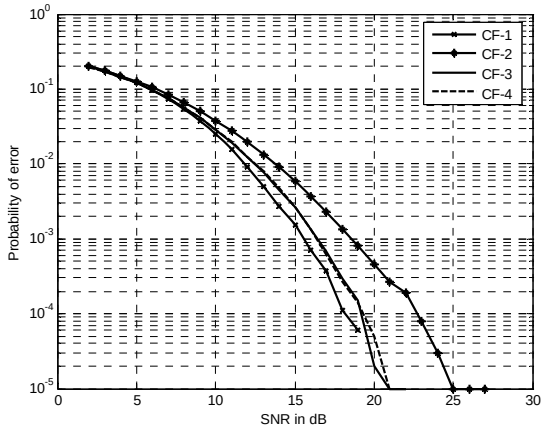
(a) 0% Outliers



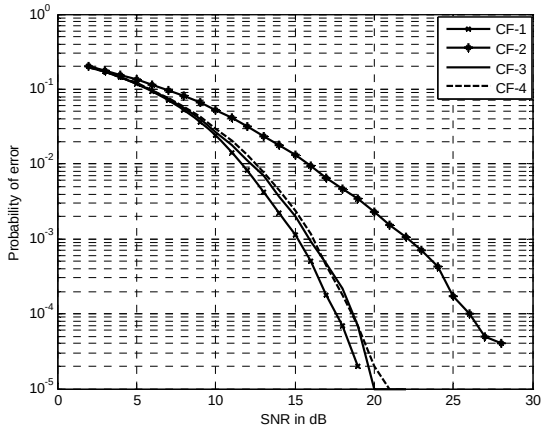
(b) 10% Outliers



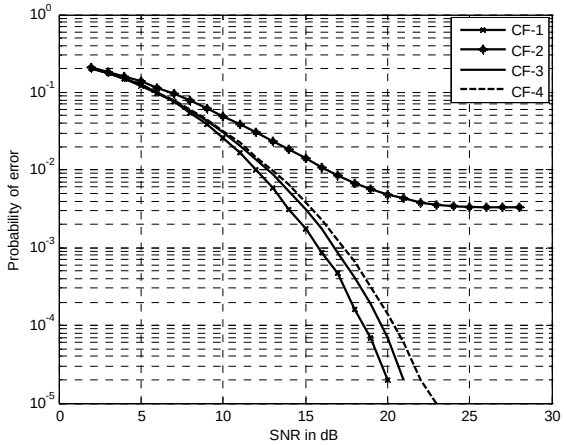
(c) 20% Outliers



(d) 30% Outliers

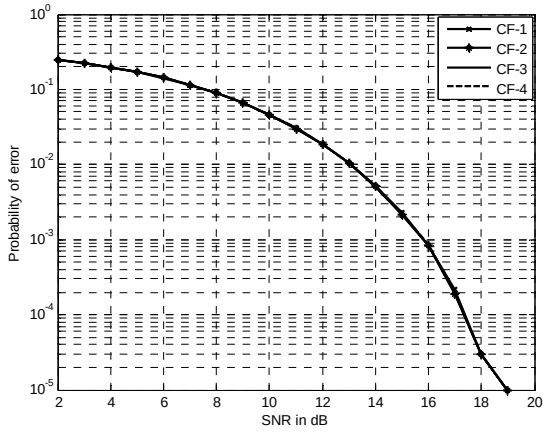


(e) 40% Outliers

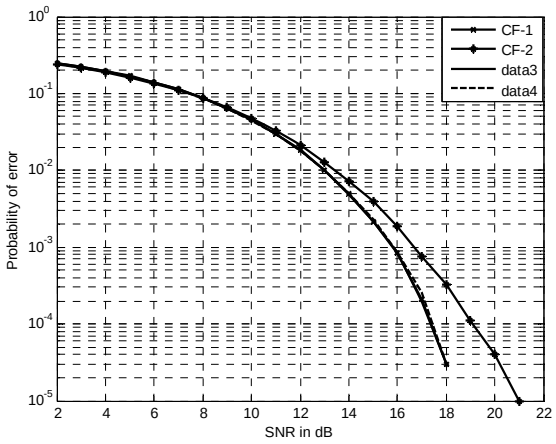


(f) 50% Outliers

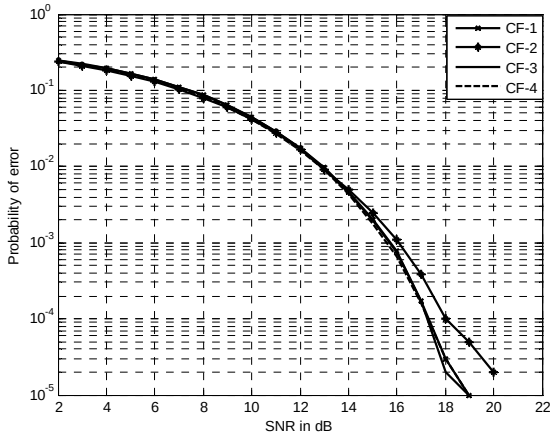
Fig. 7. 6 Comparison of BER of four different CFs based nonlinear equalizers with [.260, .930, .260] as channel coefficients and NL2



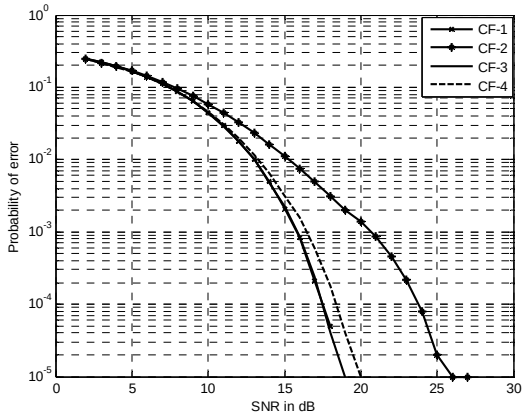
(a) 0% Outliers



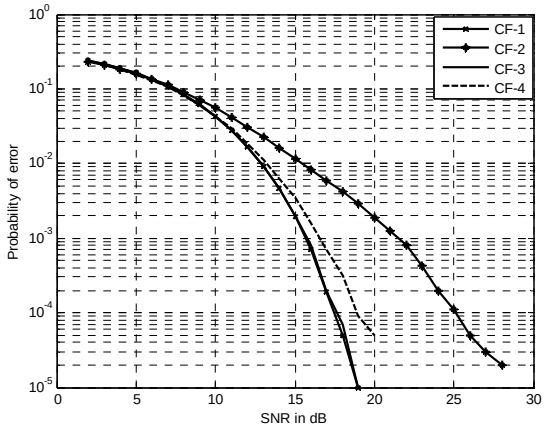
(b) 10% Outliers



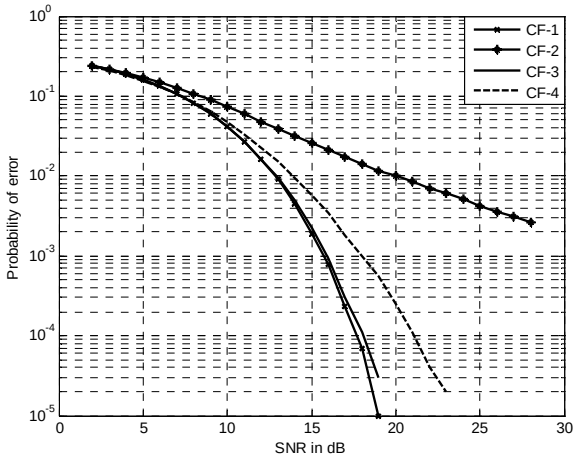
(c) 20% Outliers



(d) 30% Outliers

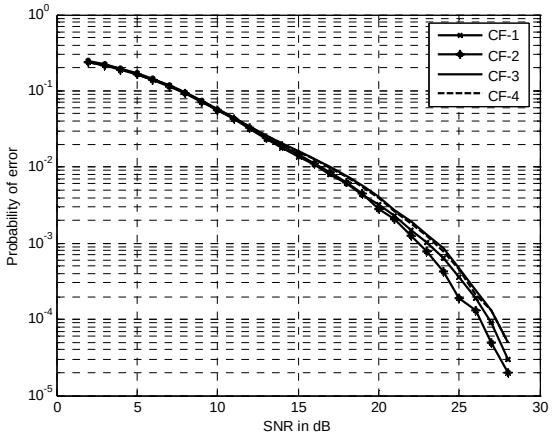


(e) 40% Outliers

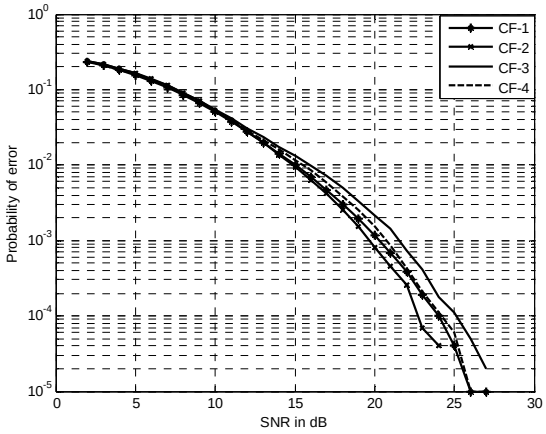


(f) 50% Outliers

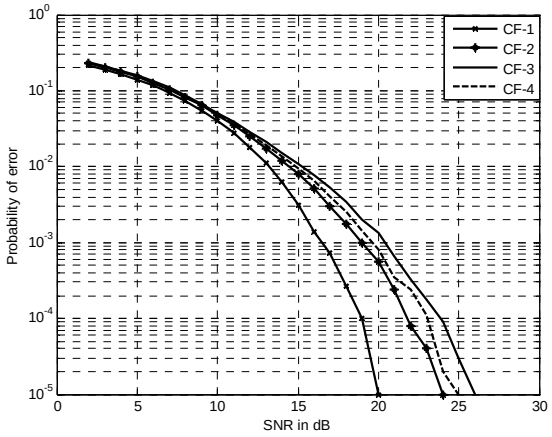
Fig. 7. 7. Comparison of BER of four different CFs based nonlinear equalizers with [.304, .903, .304] as channel coefficients and NL1



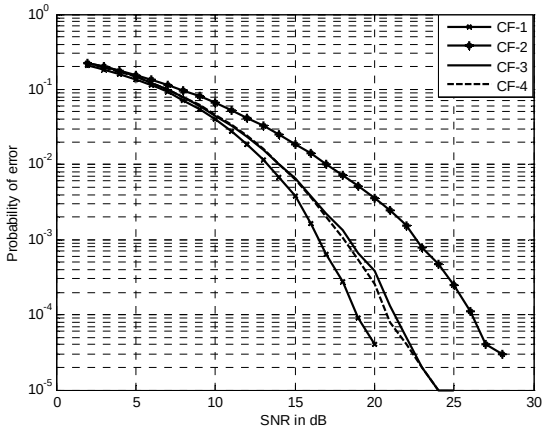
(a) 0% Outliers



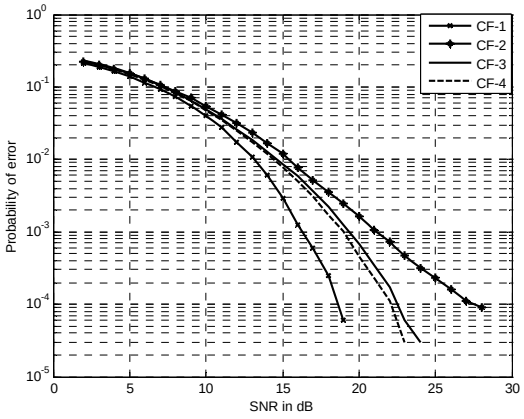
(b) 10% Outliers



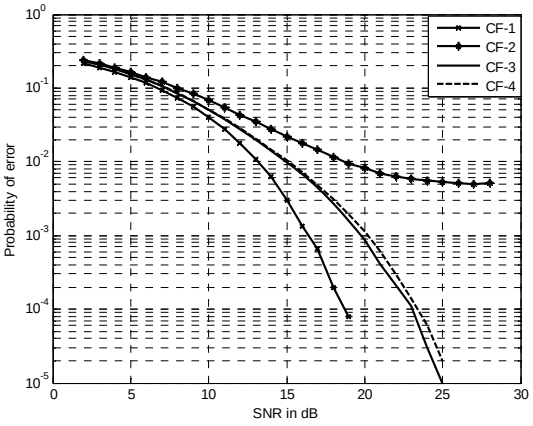
(c) 20% Outliers



(d) 30% Outliers

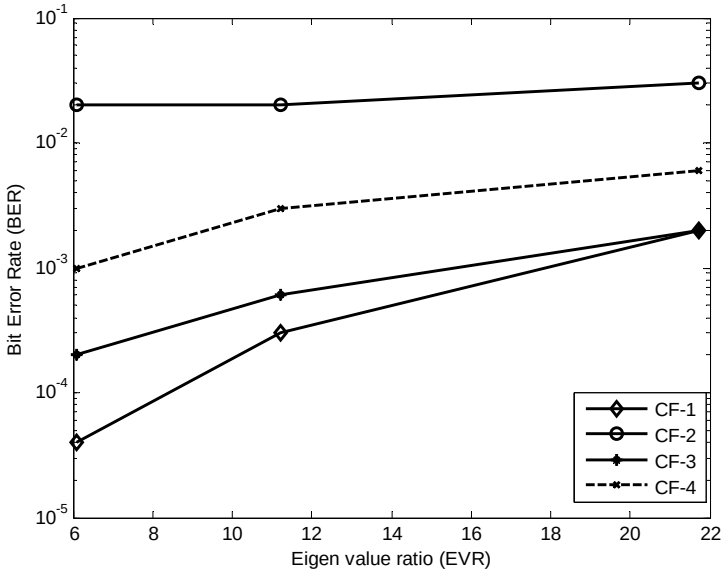


(e) 40% Outliers

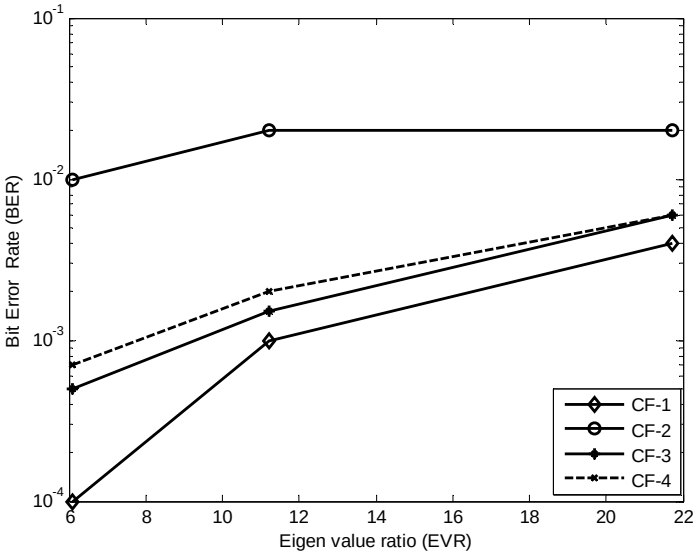


(f) 50% Outliers

Fig. 7. 8 Comparison of BER of four different CFs based nonlinear equalizers with [.304, .903, .304] as channel coefficients and NL2

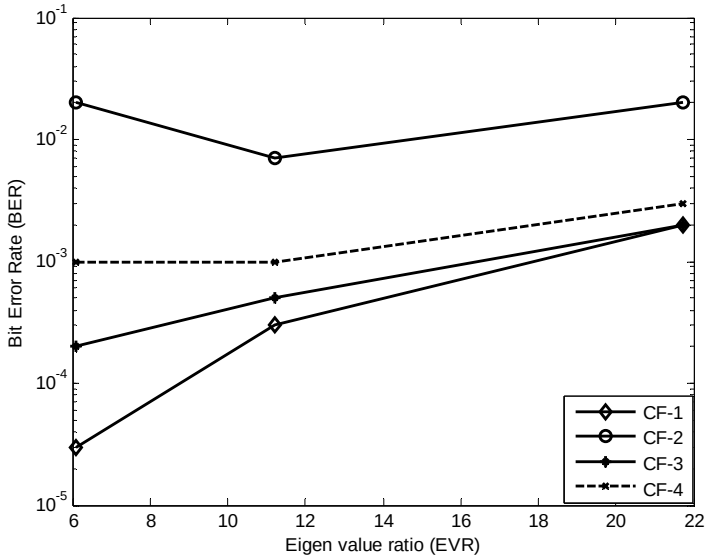


(a) NL1

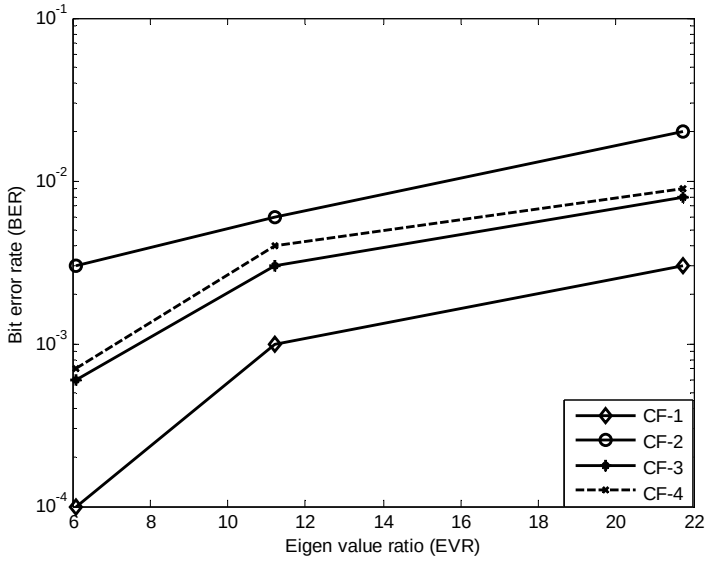


(b) NL2

Fig. 7. 9 Effect of EVR on the BER performance of the four CF-based equalizers in presence of 50% outliers



(a) NL1



(b) NL2

Fig. 7. 10 Effect of EVR on the BER performance of the four CF-based equalizers in presence of 40% outliers

7.6 Conclusion

This chapter examines and evaluates the learning capability of different norms of error when the training signal (of the inverse model) is contaminated with strong outliers. To facilitate such evaluation different nonlinear channels with varying EVRs are used. The population based BFO learning tool is developed to minimize four different norms. The robustness of these norms is assessed through simulation study. It is in general observed that the conventional squared error norm (CF2) is least robust to develop inverse models of nonlinear systems under varying noise conditions. Whereas the Wilcoxon norm (CF1) is the most robust one. In terms of robust performance, the order of the norms is CF1, CF3, CF4 and CF2.

References

[7.1] R. W. Lucky, Techniques for adaptive equalization of digital communication systems, Bell Sys.Tech. J., 45, 255-286, Feb. 1966.

[7.2] S. U. H Qureshi, Adaptive equalization, Proc. IEEE, 73(9), 1349-1387, Sept. 1985.

[7.3]E. A. Robinson and T. Durrani, Geophysical Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, 1986.

[7.4] B. Widrow and E. Walach, Adaptive Inverse Control, Prentice-Hall, Upper Saddle River, NJ, 1996.

[7.5] S. K. Nair and Jaekyun Moon, "A theoretical study of linear and nonlinear equalization in nonlinear magnetic storage channels", IEEE Trans. on neural networks, vol. 8, no. 5, pp. 1106-1118, Sept. 1997.

[7.6] H. Sun, G. Mathew and B. Farhang-Boroujeny, "Detection techniques for high density magnetic recording", IEEE Trans. on Magnetics, vol. 41, no. 3, pp. 1193-1199, March 2005.

[7.7] F. Preparata, "Holographic dispersal and recovery of Information", IEEE Trans. Inform. Theory, vol. 35, no. 5, pp. 112 -1124 , Sept., 1989.

[7.8] G. J. Gibson, S. Siu and C. F. N. Cowan, "The application of nonlinear structures to the reconstruction of binary signals", IEEE Trans. signal processing, vol. 39, no. 8, pp. 1877-1884, Aug. 1991.

[7.9] P. G. Voulgaris and C. N. Hadjicostics, "Optimal processing strategies for perfect reconstruction of binary signals under power-constrained transmission", Proc. IEEE confernec on decision and control, Atlantis, Bahamas, vol. 4, pp. 4040-4045, Dec. 2004.

[7.10] R. Touri, P. G. Voulgaris and C. N. Hadjicostis, "Time varying power limited preprocessing for perfect reconstruction of binary signals", Proc. of the 2006 American control conference, Minneapdis, USA, pp. 5722-5727, June 2006.

[7.11] J. C. Patra, R. N. Pal, R. Baliarsingh and G. Panda, "Nonlinear channel equalization for QAM signal constellation using Artificial Neural Network", IEEE Trans. on systems, man and cybernetics-Part B:cybetnetics, vol. 29, no. 2, April 1999.

[7.12] J. C. Patra, Wei Beng Poh, N. S. Chaudhari and Amitabha Das," Nonlinear channel equalization with QAM signal using Chebyshev artificial neural network", Proc. of International joint conference on neural networks, Montreal, Canada, pp. 3214-3219, August 2005.

[7.13] G. Panda, B. Majhi, D. Mohanty, A. Choubey and S. Mishra, "Development of Novel Digital Channel Equalizers using Genetic Algorithms", Proc. of National Conference on Communication (NCC-2006), IIT Delhi, pp.117-121, 27-29, January, 2006.

[7.14] K. M. Passino, "Biomimicry of Bacterial Foraging for distributed optimization and control", IEEE control system magazine, vol 22, issue 3, pp. 52-67, June 2002.

[7.15] S. Mishra, "A Hybrid least square Fuzzy bacterial foraging strategy for harmonic estimation", IEEE Trans. on Evolutionary Computation, vol 9, no. 1, pp. 61-73, Feb. 2005.

[7.16] Babita Majhi, G. Panda and A. Choubey, **"**On The Development of a new Adaptive Channel Equalizer using Bacterial Foraging Optimization Technique", Proc. of IEEE Annual India Conference (INDICON-2006), New Delhi, India, 15th-17th September, 2006, pp. 1-6.

[7.17] M. Maitra and A. Chatterjee, "A novel technique for multilevel optimal magnetic resonance brain image thresholding using bacterial foraging", Measurement, Elsevier, vol. 41, pp. 1124-1134, 2008.

[7.18] T. Y. Ji, M. S. Li, Z. Lu and Q. H. Wu, "Optimal morphological filter design using a bacterial swarming algorithm", Proc. of IEEE Congress on Evolutionary Computation (CEC 2008), Hong Kong, pp 452- 458.

[7.19] W. J. Tang, M. S. Li, Q. H. Wu and J. R. Saunders, "Bacterial foraging algorithm for optimal power flow in dynamic environments", IEEE Trans. on Circuits and Systems, vo. 55, no. 8, pp. 2433-2442, Sep. 2008.

[7.20] B. K. Panigrahi and V. Ravikumar Pandi, "Bacterial foraging optimization : Nelder-Mead hybrid algorithm for economic load dispatch", IET Gener. Transm. Distrib., vol. 2, no. 4, pp. 556-565, 2008.

[7.21] A Y. Saber and G. K. Venayagamoorthy, "Economic load dispatch using bacterial foraging technique with particle swarm optimization biased evolution", Proc. of IEEE Swarm Intelligence Symposium, St. Louis MO USA, 21-23 September, 2008, pp. 1-8.

[7.22] W. Lin, R. Liu, P. X. Liu and Max. Q-H Meng, "Parameter estimation using biologically inspired methods", Proc. of IEEE Int. conf. on Robotics and Biomimetics, Sanya, china, 15-18 Dec., 2007, pp. 1339-1343.

[7.23] D. P. Acharya, G. Panda and Y. V. S. Lakshmi, "Effect of finite register length on bacterial foraging optimization based ICA and constrained genetic algorithm based ICA algorithm", Proc. of IEEE Int. Conf. on signal processing, communication and networking, Anna Univ., Chennai, 4-6 Jan., 2008, pp. 244-249.

[7.24] M. Hanmandlu, A. V. Nath, A. C. Mishra and V. K. Madasu, "Fuzzy model based recognition of handwritten hindi numerals using bacterial foraging", Proc. of 6th IEEE Int. Conf. on computer and information science (ICIS 2007), 11-13 July 2007, pp. 309-312.

[7.25] B. Samanbabu, S. Mishra, B. K. Panigrahi and G. K. Venayagamoorthy, "Robust tuning of modern power system stabilizers using bacterial foraging algorithm", Proc. of IEEE Congress on Evolutionary Computation (CEC 2007), Singapore, 25-28 Sep. 2007, pp. 2317-2324.

[7.26] Leandro dos Santos Coelho and Camila da Costa Silveira, "Improved bacterial foraging strategy for controller optimization applied to robotic manipulator system", Proc. of IEEE Int. Symposium on intelligent control, Munich, Germany, 4-6 Oct. 2006, pp. 1276-1281.

[7.27] T. K. Das, G. K. Venayagamoorthy and U. O. Aliyu, "Bio-inspired algorithms for the design of multiple optimal power system stabilizers: SPPSO and BFA", IEEE Trans. on Industry Applications, vol. 44, no. 5, pp. 1445-1457, Sep-Oct. 2008.

[7.28] S. Mishra, C. N. Bhende and L. L Lai, "Optimization of a distribution static compensator by bacterial foraging technique", Proc. of IEEE 5th Int. Conf. on machine learning and cybernetics, Dalian, 13-16 August 2006, pp. 4075-4082.

[7.29] Ben Niu, Y Zhu, X He and X Zeng, "Optimum design of PID controllers using only a germ of intelligence", Proc. of IEEE 6th World congress on intelligent control and automation, Dalian, China, 21-23 June 2006, pp. 3584 – 3588.

[7.30] A. Ali and S. Majhi, "Design of optimum PID controller by bacterial foraging strategy", Proc. of IEEE Int. Conf. on Industrial Technology, 15-17 Dec. 2006, pp. 601-605.

[7.31] W. J. Tang, Q. H. Wu and J. R. Saunders, "Bacterial foraging algorithm for dynamic environments", Proc. of IEEE Congress on Evolutionary Computation, Canada, 16-21 July 2006, pp. 1324-1330.

[7.32] W. Lin and P. X. Liu, "Hammerstein model identification based on bacterial foraging", Electronics Letter, vol 42, no. 23, Nov. 2006.

[7.33] S. Mishra, B. K. Panigrahi and M. Tripathy, "A hybrid adaptive bacterial foraging and feedback linearization scheme based D-STATCOM", Proc. of IEEE Int. Conf. on Power system technology, Singapore , 21-24 Nov. 2004, pp. 275-280.

[7.34] Dong Hwa Kim, Ajit Abraham and Jae Hoon Cho, "A hybrid genetic algorithm and bacterial foraging approach for global optimization", Information Sciences, Elsevier, vol 177, pp. 3918-3937, 2007.

[7.35] S. Mishra and C. N. Bhende, "Bacterial foraging technique based optimized active power filter for load compensation", IEEE Trans. on Power delivery, vol. 22, no. 1, January 2007.

[7.36] M. Ulagammai, P. Venkatesh, P. S. Kannan and N. P. Padhy, "Application of bacterial foraging technique trained artificial and wavelet neural networks in load forecasting", Neurocomputing, Elsevier, vol. 70, pp. 2659-2667, 2007.

[7.37] Tai-Chen Chen, Pei-Wei Tsai, Shu-Chuan Chu and Jeng-Shyang Pan, "A novel optimization approach: Bacterial-GA foraging", Proc. of IEEE 2nd Int. Conf. on Innovative computing, information and control, 5-7 Sept. 2007, pp. 391-399.

[7.38] S. M. Dasgupta, A. Biswas, A. Abraham and S. Das, "Adaptive computational chemotaxis in bacterial foraging algorithm", Proc. of IEEE Int. Conf. on complex, intelligent and software intensive systems,4-7 March 2008, pp. 64-71.

[7.39] A. Abraham, A. Biswas, S. Dasgupta and S. Das, "Analysis of reproduction operator in bacterial foraging optimization algorithm", Proc. of IEEE World Congress on Evolutionary Computation (CEC 2008), Hong Kong, 1-6 June 2008, pp. 1476-1483.

[7.40] Joseph W. McKean, "Robust analysis of Linear models", Statistical Science, vol. 19, no. 4, pp. 562-570, 2004.

[7.41] Jer-Guang Hsieh, Yih-Lon Lin and Jyh-Horng Jeng, "Preliminary study on Wilcoxon learning machines", IEEE Trans. on neural networks, vol. 19, no. 2, pp. 201-211, Feb. 2008.

[7.42] Wei-Yen Wang, Tsu-Tian Lee, Ching-Lang Liu and Chi-Hsu Wang, "Function approximation using fuzzy neural networks with robust learning algorithm", IEEE Trans. on Systems, Man and Cybernetics-Part B : Cybernetics, vol. 27, no. 4, pp. 740-747, Aug. 1997.

[7.43] Hung-Hsu Tsai and Pao-Ta Yu, "On the optimal design of fuzzy neural networks with robust learning for function approximation", IEEE Trans. on Systems, Man and Cybernetics-Part B : Cybernetics, vol. 30, no. 1, pp. 217-223, Feb. 2000.

[7.44] J. C. Patra, A. C. Kot and G. Panda, "An intelligent pressure sensor using neural networks", IEEE Trans. on Instrumentation and Measurement, vol. 49, issue 4, pp. 829-834, Aug. 2000.

Chapter

8

Identification of Hammerstein Plants using Clonal PSO and Immunized PSO Algorithms

8.1 Introduction

PRACTICALLY it is difficult to model physical systems by mathematical analysis method. But through system identification, a suitable model can be developed which is mathematically equivalent to a given physical system. Many practical systems possess inherent nonlinear characteristics due to harmonic generation, intermediation, desensitization, gain expansion and chaos. Identification of such nonlinear complex plants plays a significant role in analysis and design of control systems. The Hammerstein model is widely used because its structure describes the nonlinearity associated with practical dynamic systems. Several methods have been proposed in the literature for identification of Hammerstein model by using correlation theory [8.1], orthogonal functions [8.2], polynomial functions [8.3], piecewise linear model [8.4], artificial

neural networks [8.5], genetic algorithm [8.6], radial basis function (RBF) networks [8.7], particle swarm optimization (PSO) [8.8, 8.9] and bacterial foraging optimization (BFO) [8.10] techniques . The Particle Swarm Optimization (PSO) was developed by Eberhart and Kennedy in 1995 [8.11] inspired by swarm intelligence theory such as birds flocking, fish schooling etc. It gained a lot of attention in various optimal control system applications because of its faster convergence [8.12], reduced memory requirement, lower computational complexity and easier implementation as compared to other evolutionary algorithms. However, there are some problems associated with the basic PSO, such as premature convergence and stagnation at the local optimal solution. In [8.13] it is shown that the PSO performs well in early generations than any other evolutionary algorithm, but it degrades as the number of generations increases. Therefore it has a slow fine tuning ability of the solution. Several studies have been made to improve the performance of PSO [8.14] - [8.17].

The biological immune system (BIS) is a multilayer protection system where each layer provides different types of defense mechanisms for detection, recognition and responses. It also resists infectious diseases and reacts to foreign substances. Following the principle of BIS a new tool of computational intelligence known as artificial immune system (AIS) [8.18]- [8.20] has evolved which finds applications in optimization problems [8.21, 8.22], computer security [8.23, 8.24], fault detection [8.25, 8.26], job scheduling [8.27] and clustering. The four forms of AIS algorithm reported in the literature are immune network model [8.19], negative selection [8.23, 8.28], clonal selection [8.21, 8.22] and danger theory [8.29].

In this chapter two new hybrid algorithms known as Clonal PSO (CPSO) and Immunized PSO (IPSO) have been proposed by suitably combining the good features of PSO and AIS algorithms. The performance of these new algorithms has been assessed by employing them in identification of various standard Hammerstein models. The nonlinear static part of the model is represented by a single layer low complexity nonlinear functional link artificial neural network (FLANN) architecture [8.30, 8.31]. The weights of the FLANN structure and the dynamic part of the model are estimated by the proposed algorithms.

8.2 Identification of Hammerstein plants using FLANN

8. 2. 1 Hammerstein model

The nonlinear dynamic system described by Hammerstein model is composed of a nonlinear static block in series with a linear dynamic system block as shown in Fig.8.1.

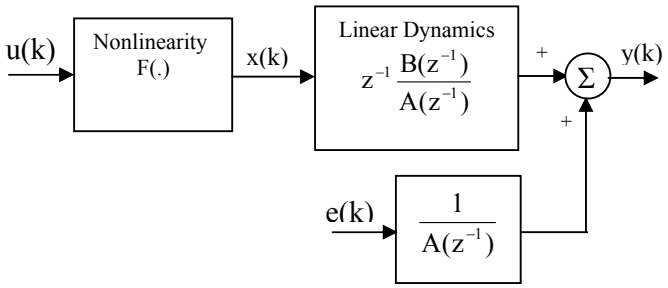


Fig. 8.1 The Hammerstein Model

The model in general is described by

$$A(z^{-1})y(k) = B(z^{-1})x(k-1) + e(k) \tag{8.1}$$

$$x(k) = F(u(k)) \tag{8.2}$$

$$A(z^{-1}) = 1 + a_1 z^{-1} + ... + a_n z^{-n} \tag{8.3}$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} + ... + b_r z^{-r} \tag{8.4}$$

where z^{-1} denotes an unit delay. In this model $u(k), y(k)$ and $e(k)$ represent the input, output, and noise samples at instant k respectively. The intermediate signal $x(k)$ is not accessible for measurement. The symbols n and r are known degrees of polynomials of $A(z^{-1})$ and $B(z^{-1})$ respectively. The function $F(.)$ is assumed to be nonlinear and unknown.

The objective of the identification task of the Hammerstein model is to determine the system parameters $\{a_i\}, \{b_j\}$ of the linear dynamic part and response matching at the nonlinear static part and the output of the model using known input and output samples $u(k)$ and $y(k)$.

8. 2. 2 FLANN architecture for modeling nonlinear static part

In this Chapter, the nonlinear static part of the Hammerstein model is represented by a FLANN structure. Its input signal $u(k)$ at the k th instant is functionally expanded to a number of nonlinear values to feed to an adaptive linear combiner whose weights are altered according to an iterative learning rule. The types of expansion suggested in the literature are either trigonometric, power series or Chebyshev expansion. For trigonometric expansion the linear matrix is given by

$$\Phi_i\{u(k)\} = \begin{cases} 1 & \text{for } i = 0 \\ u(k) & \text{for } i = 1 \\ \sin(i\pi u(k)) & \text{for } i = 2,4,....M \\ \cos(i\pi u(k)) & \text{for } i = 3,5,...M+1 \end{cases} \tag{8.5}$$

where $i = 1, 2, ..., M/2$. As a result the total expanded values including an unity bias input become $2M+2$. Let the corresponding weight vector be represented as $w_i(k)$ having $2M+2$ elements. The estimated output of the nonlinear static part as shown in Fig. 8.2 is given by

$$F(u(k)) = \sum_{i=1}^{2M+2} w_i \Phi_i(u(k)) + \varepsilon(k) \tag{8.6}$$

where $\varepsilon(k)$ is approximation error.

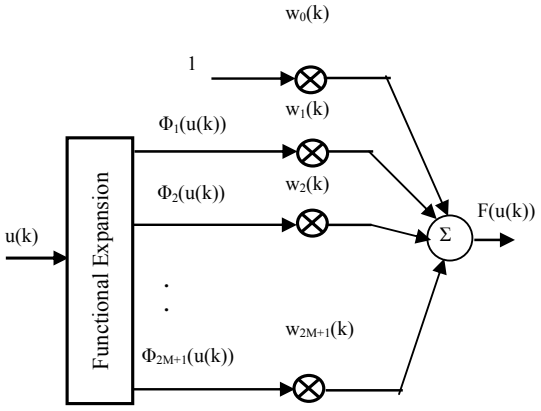


Fig. 8.2 Structure of FLANN model

Substitution of (8.4) in (8.1) gives

$$A(z^{-1})y(k) = [b_0 F(u(k-1)) + b_1 F(u(k-2)) + ... + b_r F(u(k-r-1))] + e(k) \tag{8.7}$$

Similarly from (8.6) and (8.7) we get

$$A(z^{-1})y(k) = [b_0\left(\sum_{i=1}^{2M+2} w_i \Phi_i(u(k-1)) + \varepsilon(k-1)\right) + ...$$

$$.. + b_r\left(\sum_{i=1}^{2M+2} w_i \Phi_i(u(k-r-1)) + \varepsilon(k-r-1)\right)] + e(k) \tag{8.8}$$

Rearrangement of (8.8) gives

$$A(z^{-1})y(k) = [\sum_{i=0}^{r} b_i w_1 \Phi_1(u(k-i-1)) + ..$$

$$.. + \sum_{i=0}^{r} b_i w_{2M+2} \Phi_{2M+2}(u(k-i-1))$$

$$+ \sum_{i=0}^{r} b_i \varepsilon(k-i-1)] + e(k) \tag{8.9}$$

The identification structure of Hammerstein model corresponding to (8.9) is shown in Fig. 8.3.

Here $v(k)$ and θ represented as

$$v(k) = e(k) + \sum_{i=0}^{r} b_i \varepsilon(k-i-1) \tag{8.10}$$

$$\theta = [\theta_a^T, \theta_{w1}^T, ...\theta_{w_{2M+2}}^T]^T \tag{8.11}$$

where

$$\theta_a^T = [a_1, a_2, ...a_n]^T \tag{8.12}$$

$$\theta_{w_i} = [\theta_{w_i}(1), \theta_{w_i}(2), .. \theta_{w_i}(r+1)]^T$$

$$= [b_0 w_i, b_1 w_i, ..., b_r w_i]^T \tag{8.13}$$

At the k th instant $\varphi(k)$ is given by

$$\varphi(k) = [\varphi_a^T(k), \varphi_{w1}^T(k), ...\varphi_{w_{2M+2}}^T(k)]^T \tag{8.14}$$

where

$$\varphi_a(k) = [-y(k-1), -y(k-2), ... - y(k-n)]^T \tag{8.15}$$

$$\varphi_{w_i}(k) = [\Phi_i(u(k-1)), ...\Phi_i(u(k-r-1))]^T \tag{8.16}$$

Using (8.10)-(8.11) and (8.14) , (8.8) can be expressed as

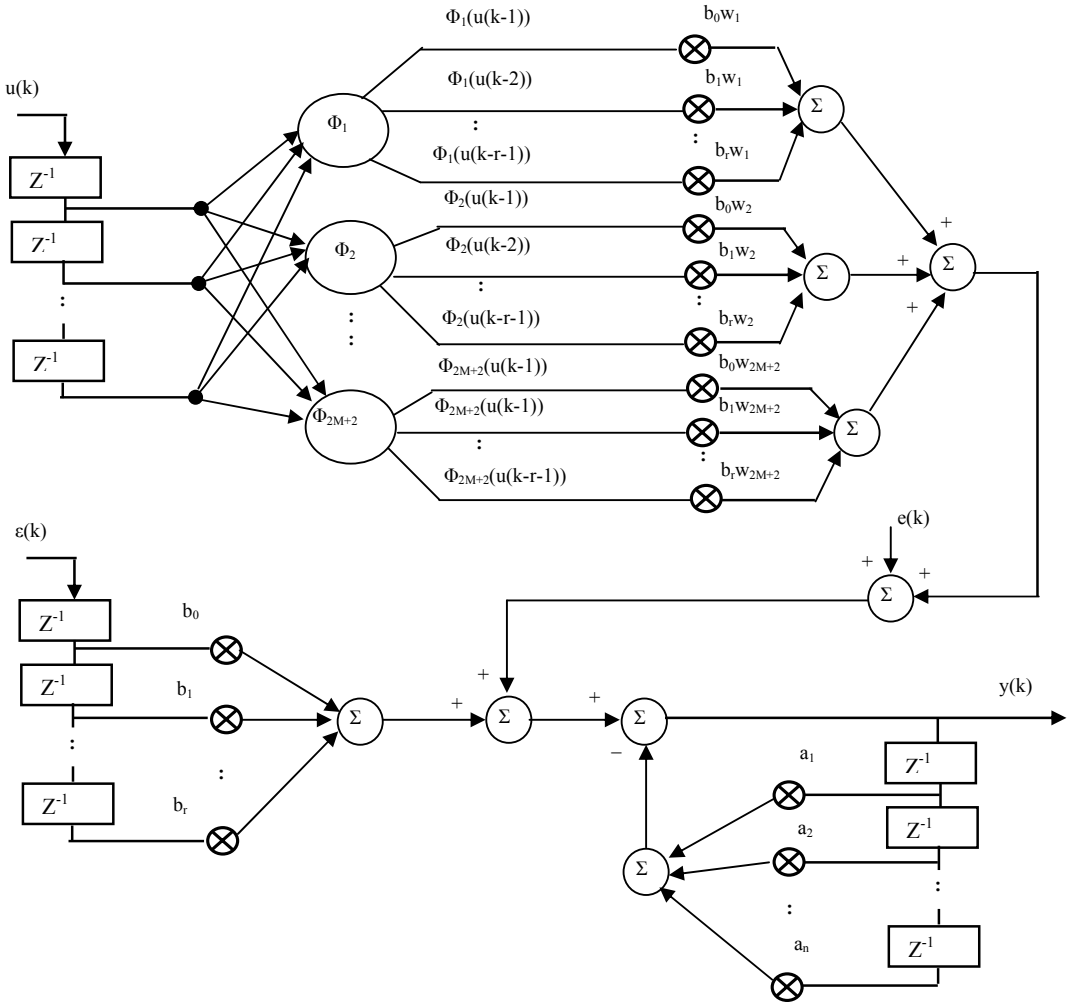


Fig. 8. 3 Adaptive Identification model of the generalized Hammerstein Plant

$$y(k) = \varphi^T(k)\theta + v(k) \tag{8.17}$$

The objective here is to estimate the system parameters defined in (8.11). If derivative based least square method is taken then the estimate of these system parameters is given by

$$\overset{\Lambda}{\theta} = \left[\sum_{k=N_s+1}^{N_s+N} \varphi(k)\varphi^T(k)\right]^{-1}\left[\sum_{k=N_s+1}^{N_s+N} \varphi(k)y(k)\right] \tag{8.18}$$

where N is the number of input output data. Substituting $\hat{w}_1 = 1$ the parameters of linear dynamic part are estimated as $\hat{a}_1,.....\hat{a}_n,\hat{b}_1,.....\hat{b}_r$. Equation (8.18) provides an estimate of the parameters which involves matrix inversion and hence computationally very expensive.

Further the input in this case is obtained from an nonlinear model. The conventional derivative based method to estimate the pole-zero parameters often leads to instability during training. Hence it is motivating to devise efficient and reliable methods to efficiently identify Hammerstein plant using two new population based CPSO and IPSO algorithms.

8.3 Proposed clonal PSO and immunized PSO algorithms

In PSO algorithm, a swarm consists of a set of volume-less particles (a point) moving in a D-dimensional search space, each representing a potential solution. The i th particle is represented by a vector: $X_i = [x_{i1}, x_{i2} \ldots . x_{id} \ldots . x_{iD}]$ .The best previous position (the position giving the best fitness value) of the i th particle is recorded and represented as $P_i = [p_{i1}, p_{i2} \ldots . p_{id} \ldots . p_{iD}]$. At each iteration, the global best particle in the swarm is represented by $P_g = [p_{g1}, p_{g2} \ldots . p_{gd} \ldots . . p_{gD}]$ .The velocity of the i th particle is represented as $V_i = [v_{i1}, v_{i2} \ldots v_{id} \ldots v_{iD}]$. The maximum velocity and the range of particles are given by $V_{\max} = [v_{\max 1}, v_{\max 2} \ldots . . v_{\max d} \ldots . . . . v_{\max D}]$ and $X_{\max} = [x_{\max 1}, x_{\max 2} \ldots . . x_{\max d} \ldots . . . . x_{\max D}]$. The velocity and position of the d th element of the i th particle at $(k+1)$ th search from the knowledge of previous search are modified according to (8.19)-(8.22).

$$V_{id}(k+1) = w(k) * v_{id}(k) + c_1 * r_1 * (p_{id}(k) - x_{id}(k)) + c_2 * r_2 * (p_{gd}(k) - x_{id}(k)) \quad (8.19)$$

$$V_{id}(k+1) = \begin{cases} v_{\max d}, & v_{id}(k+1) > v_{\max d} \\ -v_{\max d}, & v_{id}(k+1) < -v_{\max d} \end{cases} \quad (8.20)$$

$$X_{id}(k+1) = X_{id}(k) + V_{id}(k+1) \quad (8.21)$$

$$X_{id}(k+1) = \begin{cases} x_{\max d}, & x_{id}(k+1) > x_{\max d} \\ -x_{\max d}, & x_{id}(k+1) < -x_{\max d} \end{cases} \quad (8.22)$$

where $i = 1, 2, \ldots . . N_1, d = 1, 2, \ldots . . . . D$ and N_1 is the number of particles . The symbols r_1 and r_2 represent random numbers between 0 and 1, c_1 and c_2 denote acceleration constants. The inertia weight, w is employed to control the impact of pervious history of velocities on the current one in order for tradeoff between the global and local exploitations and is given by [8.16].

$$w(k) = w_0 - \frac{(w_0 - w_1) * k}{itr} \tag{8.23}$$

where k =generation counter (from 1 to itr)

itr =number of iterations

$w_0 = 0.9$ and $w_1 = 0.4$

8. 3. 1. The CPSO Algorithm

In conventional PSO, the velocity of each particle in the next search is updated using the knowledge of its past velocity, personal and global best positions. Since the global best position after a search is the best among all personal best positions, their use in updating the velocity has little contribution in moving to new positions. Therefore in the present investigation second term in the velocity update equation (8.19) of conventional PSO is not considered.

Further according to clonal selection principle when an antigen or a pathogen invades the organism, a number of antibodies is produced by the immune cells. The fittest antibody undergoes cloning operation to produce number of new cells. These are used to eliminate the invading antigens. Employing this principle of AIS in PSO it is proposed here that each particle moves to the global best position after a search is complete wherefrom each individual starts its next search. The above idea implies that during any k th search the position of the d th element of the i th particle becomes equal to the global best position i.e. $x_{id}(k) = p_{gd}(k)$. As a result the third term of the velocity updates equation (8.19) of conventional PSO becomes zero. Incorporating the above two ideas into the conventional PSO algorithm leads to a simplified velocity update equation

$$V_{id}^{'}(k+1) = w(k) * v_{id}^{'}(k) \tag{8.24}$$

$$X_{id}^{'}(k+1) = X_{id}^{'}(k) + V_{id}^{'}(k+1) \tag{8.25}$$

where $i = 1,2 \ldots . N_1, d = 1,2 \ldots . D$, The inertia weight $w(k)$ is computed according to (8.23). According to this algorithm after every search all particles migrate to the global best position wherefrom each particle disperses again according to individual's magnitude and direction of velocity. The same process is repeated until the position of gbest finally represents the optimal solution of the problem.

8. 3. 2 The IPSO algorithm

The CPSO algorithm is relatively simpler than conventional PSO and performs satisfactorily when applied to different optimization problems. However one important observation in this new algorithm is that computation of every new position of a particle depends on two factors: i.e. time varying inertia weight w(k) and its initial velocity. As a result the diversification in the solution space after each search becomes limited. Hence there is a chance that the final solution in this approach might lead to a local one. To overcome this shortcoming another new algorithm called immunized PSO algorithm is proposed by introducing mutation process into the algorithm.

In this case, like the CPSO algorithm, each particle after a search occupies the global best position. Then the mutation operation is carried out on the position vector of the particles to enable random diversifications of their positions. Since the position of each particle is changed unlike in CPSO, the third term remains. But the second term which contributes to change in velocity due to local best is not used. Thus the update equation becomes

$$V_{id}^{"}(k+1) = w(k) * v_{id}^{"}(k) + c_2^{"} * r_2^{"} * (p_{gd}(k) - x_{id}(k)) \tag{8.26}$$

$$X_{id}^{"}(k+1) = X_{id}^{"}(k) + V_{id}^{"}(k+1) \tag{8.27}$$

From among the updated positions of the particles the global best position is selected and then cloned. Then the cloned cells undergo a mutation mechanism by following the hyper mutation concept of AIS [8.21, 8.22]. The mutation operation has fine-tuning capabilities which helps to achieve better optimal solution. The single dimension mutation (SDM) operation [8.16] is defined as

$$xm_{id}(k+1) = x_{T_1 d}(k+1) + 0.01 * x_{T_2 d}(k+1) \tag{8.28}$$

$$xm_{(i+1)d}(k+1) = x_{T_2 d}(k+1) + 0.01 * x_{T_1 d}(k+1) \tag{8.29}$$

where T_1 and T_2 represent the particles' positions to be mutated and are chosen randomly from the set of cloned positions. In order to increase the efficiency of mutation an adaptive SDM (ASDM) is also proposed where the constants 0.01 is replaced by a parameter z whose values varies with the number of search. The value of $z(k)$ at k th search is given by

$$z(k) = (z_i - z_f)\left(\frac{I-k}{I}\right) + z_f \tag{8.30}$$

where z_i and z_f are initial and final values of z and are selected within the range [0,1]. The symbol I represents the maximum number of search. The fitness values of updated position as well as the mutated position of particles are then evaluated and the overall best location is selected for evaluation. In the next search the best location is again cloned and the process continues. The IPSO introduces improved search of particles in the D-dimensional space using mutation.

8.4 Weight update of the Hammerstein model

8.4.1 Identification algorithm using FLANN structure and PSO based training

Step 1. Determination of output of the Hammerstein Model:

Uniformly distributed random $'k'$ samples are generated to act as input during training. These are passed through the nonlinear static part and subsequently though the linear dynamic part of the Hammerstein model to produce output $y(k)$.

Step 2. Functional expansion of input:

The same input samples are also passed through the model consisting FLANN structure. Each input sample undergoes either trigonometric expansion as illustrated in (8.5), power series or square cube expansion.

Step 3. Initialization of positions and velocities of swarm:

The weight vector of the Hammerstein model of Fig.8.3 is considered as a particle. Similar to other evolutionary algorithms a set of particles representing a set of initial solutions is chosen. The weight vector of D elements comprises of $(M+2)$ number of elements for FLANN along as well as $(n+r)$ elements for linear dynamic part. Each weight vector consists of $D = M + n + r + 2$ weights which are each initialized as random numbers. For the i th particle, the position vector (which represents the weight vector) is given by

$$W_i = X_i = [w_{i1}\, w_{i2} \ldots\ldots w_{id} \ldots\ldots w_{iD}] \tag{8.31}$$

where w_{id} is the d th weight of the i th particle. Similarly the velocity assigned to the i th particle is expressed as

$$V_i = [v_{i1}\, v_{i2}\, ...v_{id}\, ...v_{iD}] \tag{8.32}$$

Initially the personal best position each i th particle achieved is same as the initial i th weight vector W_i and is represented as

$$P_i = W_i = [w_{i1}\, w_{i2}\, ...w_{id}\, ...w_{iD}] \tag{8.33}$$

Step 4. Calculation of output of model:

The output of model is computed using FLANN model according to (8.17) and the weight vector defined in (8.13).

Step 5. Fitness Evaluation:

The output of the model $\hat{y}_i(k)$ due to k th sample and i th particle is compared with the output of the plant to produce error signal given by

$$e_i(k) = y_i(k) - \hat{y}_i(k) \tag{8.34}$$

For each i th weight vector the mean square error (E) is determined and is used as the fitness function given by

$$E(i) = \frac{\sum_{k=1}^{K} e_i^2(k)}{K} \tag{8.35}$$

The identification task is then reduces to a minimization of the MSE defined in (8.35) using PSO and new algorithms.

Step 6. Updation:

The velocity and the position of the d th weight of each i th particle for the next search are obtained by the update rule given in eqs. (8.19)-(8.23).

Step 7. Evaluation of global best position of particle:

The fitness values of all particles are evaluated following step5. The best fitness value that is, the minimum MSE (MMSE) is obtained and its corresponding D weights are identified and termed as the global best. It is denoted by

$$P_g = W_g = [w_{g1}\, w_{g2}\, ...w_{gd}\, ...w_{gD}] \tag{8.36}$$

Step 8. Stopping Criteria:

The search process described in steps 1 to 6 continues until all the particles in the swarm (the weight vectors) have achieved the global best position corresponding to a predefined mean square error.

8.4.2 Identification algorithm using FLANN structure and CPSO based training

In CPSO, steps 1 to 5 of basic PSO remain the same .

Step 6. Updation:

The position and velocity of d th weight of each i th particle for the next search is obtained by the update rule given in (8.24)-(8.25). The maximum velocity and position of particles after updating is controlled by (8.20) and (8.22).

Step 7. Evaluation of global best position of particle:

The global best position of particles is evaluated in similar way as described in step 7 of PSO.

Step 8. Cloning Operation:

The global best position is cloned in the sense that all particles of the swarm start their next search from this position..

Step 9. Stopping Criteria:

The search process of steps 4 to 8 continues until all the particles in the swarm (the weight vectors) have attained the global best corresponding to a predefined mean square error.

8.4.3. Identification algorithm using FLANN structure and IPSO based training

Steps 1 to 5 are same as those of CPSO.

Step 6. Updation:

The position and velocity of d th weight of each i th particle for the next search is obtained by the update rule given (8.26)-(8.27). The maximum velocity and position of particles after updating are governed by (8.20) and (8.22).

Step 7. Evaluation of global best position of particle:

The global best position of particles is evaluated in similar way following step 7 of PSO algorithm.

Step 8. Cloning Operation:

The global best particle position is cloned so that all particles occupy the same best position.

Step 9. Mutation:

Mutation process is incorporated to introduce variations in the cloned position. Probability of mutation P_m is taken to be greater than 0.5. The mutated children produced are given by

$$xm_{id}(k+1) = x_{T_1 d}(k+1) + z(k) * x_{T_2 d}(k+1) \tag{8.37}$$

$$xm_{(i+1)d}(k+1) = x_{T_2 d}(k+1) + z(k) * x_{T_1 d}(k+1) \tag{8.38}$$

Step 10. Stopping Criteria:

The search process from described in steps 4 to 9 continues until all the particles in the swarm (the weight vectors) have converged to the global best position yielding a predefined minimum mean square error.

8.5 Simulation study

To demonstrate the improved identification performance of the two new algorithms simulation study using MATLAB is carried out. Four standard Hammerstein plants are used for identification using CPSO and IPSO algorithms. The accuracy of identification of the proposed models are assessed by comparing the following results.

1. True and estimated responses at the output of nonlinear static part.
2. The true and estimated coefficients of the linear dynamic part.
3. Comparison of sum of squared errors (SSE) between true and overall estimated responses. The sum of squared error is defined as

$$SSE(k) = \sum_{k=1}^{K} (y(k) - \hat{y}(k))^2 \tag{8.39}$$

where y(k) is true output and $\hat{y}(k)$ is estimated output during testing.

Example 1

The Hammerstein plant used for identification [8.32] is given by

$$A(z^{-1})y(k) = B(z^{-1})x(k-1) + e(k) \tag{8.40}$$

$$x(k) = F(u(k)) = u(k) + 0.5u^3(k) \tag{8.41}$$

$$A(z^{-1}) = 1 + 0.8z^{-1} + 0.6z^{-2} \tag{8.42}$$

$$B(z^{-1}) = 0.4 + 0.2z^{-1} \tag{8.43}$$

The input to the plant and the identification model is an uniformly distributed input lying between [-3.0, 3.0]. A zero mean white Gaussian noise with standard deviation of 0.01 is added to the plant. The number of input samples used to train the network is 300. In the model the nonlinear static part is represented by a FLANN structure. Each input sample is expanded to four terms by power series expansion as

$$\Phi_i\{u(k)\} = \begin{cases} 1 & for\ \ i = 0 \\ u(k) & for\ \ i = 1 \\ u^i(k) & for\ \ i = 2,3 \end{cases} \tag{8.44}$$

These expanded inputs are weighted by the coefficient of the FLANN. The weights are updated by using GA, CLONAL, PSO, CPSO and IPSO algorithms. In all cases the initial population of particles is taken as 70. The weights of the model are trained for 40 generations. The positions of the particles are considered within range [-2, 2] and their velocities are limited within the range [-1.5, 1.5]. In case of IPSO the probability of mutation P_m is taken as 0.8. The values of z_i and z_f are set at 0.9 and 0.05 respectively. The true and estimated outputs of nonlinear static part of the plant are compared in Fig.8.4. Comparison of estimates of the system parameters of linear dynamic part are shown in Table 8.1 along with percentage of error within pair of brackets.

$$\text{Percentage of error} = \frac{\text{Actual Coefficient} - \text{Estimated Coeffcient}}{\text{Actual Coefficient}} \tag{8.45}$$

The CPU time required for training of the model is presented in Table 8.2. The comparative results of sum of squared errors (SSE) obtained during testing is presented in Table 8.2.

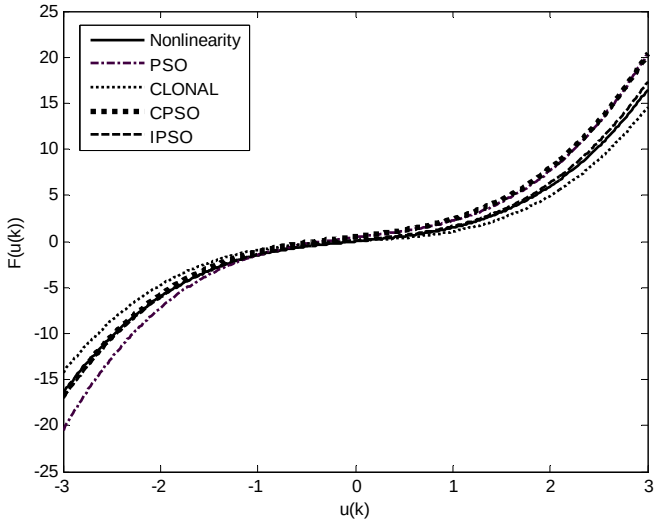


Fig.8.4 Comparison of response at the output of nonlinear static part of the plant and the corresponding models of Example 1

Table 8.1

Comparison of true and estimated parameters of system for dynamic part of the model of Example 1

Parameters	True Values	Estimated Values				
		IPSO	CPSO	PSO	CLONAL	GA
a_1	0.8	0.805 (0.6 %)	0.900 (12.5 %)	0.850 (6.2 %)	0.826 (3.2 %)	0.835 (4.3 %)
a_2	0.6	0.590 (1.6 %)	0.606 (1.0 %)	0.650 (8.3 %)	0.614 (2.3 %)	0.630 (5.0 %)
b_0	0.4	0.409 (2.2 %)	0.350 (12.5 %)	0.336 (16.0 %)	0.456 (14.0 %)	0.345 (13.7 %)
b_1	0.2	0.210 (5.0 %)	0.240 (20.0 %)	0.240 (20.0 %)	0.236(18.0 %)	0.240 (20.0 %)

Table 8.2

Comparison of CPU time and SSE for identifying the plant of Example 1

Algorithm	During Training CPU Time (Sec.)	During Testing SSE
IPSO	28.468	0.661
CPSO	28.448	4.334
PSO	27.813	10.683
CLONAL	35.125	1.948
GA	41.213	7.358

Example 2

In this example [8.32] the plant is described by

$$A(z^{-1})y(k) = B(z^{-1})x(k-1) + e(k) \tag{8.46}$$

$$x(k) = F(u(k))$$

$$= \begin{cases} -2.0 & (-3.0 \le u(k) < -1.8) \\ u(k)/0.6 + 1.0 & (-1.8 \le u(k) < -0.6) \\ 0.0 & (-0.6 \le u(k) < 0.6) \\ u(k)/0.6 - 1.0 & (0.6 \le u(k) < 1.8) \\ 2.0 & (1.8 \le u(k) \le 3.0) \end{cases} \tag{8.47}$$

$$A(z^{-1}) = 1 + 0.8z^{-1} + 0.6z^{-2} \tag{8.48}$$

$$B(z^{-1}) = 0.4 + 0.2z^{-1} \tag{8.49}$$

This system has saturation and dead- zone nonlinearity. The input signal is a uniformly distributed signal lying between [-3.0, 3.0] and a total of 100 such samples are used for training. Zero mean white Gaussian noise, $e(k)$ is with standard deviation 0.01.In the FLANN, each input sample of the FLANN is expanded as

$$\Phi_i\{u(k)\} = \begin{cases} 1 & for\ i = 0 \\ u(k) & for\ i = 1 \\ \sin(u(k)) & for\ i = 2 \end{cases} \tag{8.50}$$

The initial population of particles is taken as 70. The weights of the model are trained for 40 generations. The positions of the birds are considered within the range [-2, 2] and their corresponding velocities are taken within the range [-1.5, 1.5].In case of IPSO, the probability of mutation P_m is taken as 0.8. The values of z_i and z_f are 0.9 and 0.05 respectively. The true and estimated output of nonlinear static part of the model is presented in Fig. 8.5. Comparison of true and estimated of the system parameters along with percentage of error of linear dynamic part shown in Table 8.3. Table 8.4 shows the comparative result of CPU time required for training of model and sum of squared errors (SSE) obtained during testing.

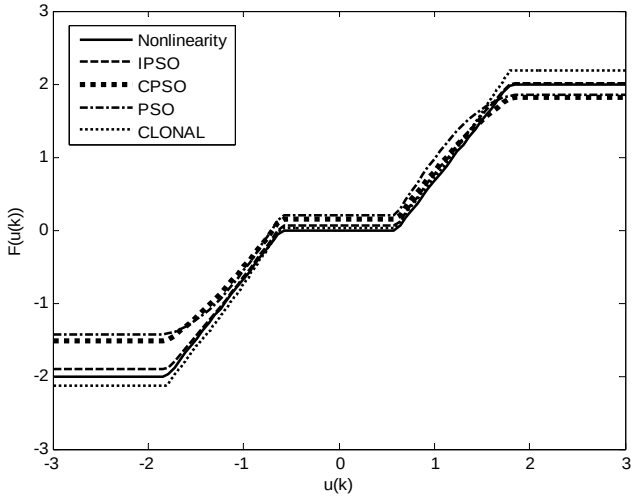


Fig. 8.5 Comparison of response at the output of nonlinear static part of the plant and the corresponding models of Example 2

Table 8.3

Comparative results of estimates of system parameters for dynamic part of the model of Example 2

Parameters	True Values	Estimated Values IPSO	CPSO	PSO	CLONAL	GA
a_1	0.8	0.810 (1.2 %)	0.690 (13.7 %)	0.900 (12.5 %)	0.752 (6.0 %)	0.861(7.62 %)
a_2	0.6	0.600 (0.0 %)	0.520 (13.3 %)	0.553(7.83 %)	0.582 (3.0 %)	0.563 (6.17 %)
b_0	0.4	0.410 (2.5 %)	0.450 (12.5 %)	0.466(16.5 %)	0.378(5.5 %)	0.441 (10.2 %)
b_1	0.2	0.200 (0.0 %)	0.220 (10.0 %)	0.251 (25.5 %)	0.160(20.0 %)	0.236(18.0 %)

Table 8.4

Comparison of CPU time and SSE for identifying the plant of Example 2

Algorithm	During Training CPU Time (Sec.)	During Testing SSE
IPSO	9.844	0.149
CPSO	9.687	0.513
PSO	9.578	2.410
CLONAL	14.906	0.486
GA	19.104	1.231

Example 3

The third Hammerstein model [8.10] chosen for identification is

$$A(z^{-1})y(k) = B(z^{-1})x(k-1) + e(k) \tag{8.51}$$

$$x(k) = F(u(k))$$

$$= u(k) + 0.5u^2(k) + 0.3u^3(k) + 0.1u^4(k) \tag{8.52}$$

$$A(z^{-1}) = 1 + 0.9z^{-1} + 0.15z^{-2} + 0.02z^{-3} \tag{8.53}$$

$$B(z^{-1}) = 0.7 + 1.5z^{-1} \tag{8.54}$$

Fig.8.3 represents the plant model. is taken into consideration. The input to this model is a uniformly distributed signal lying between [-1.0, 1.0].The number of input sample used during training is 300. The white Gaussian noise with zero mean and standard deviation 0.01 is added at the output of the plant. In FLANN each input sample is expanded to 5 terms

$$\Phi_i\{u(k)\} = \begin{cases} 1 & for\ i = 0 \\ u(k) & for\ i = 1 \\ \sin((i-1)*u(k)) & for\ i = 2,4 \\ \cos((i-1)*u(k)) & for\ i = 3 \end{cases} \tag{8.55}$$

The initial population of particles is taken as 90. The weights of the model are trained for 40 generations. The positions of the particles are considered within range [-2, 2] and their

velocities are taken in the range [-1.5, 1.5]. In case of IPSO the probability of mutation P_m is taken as 0.8. The values of z_i and z_f used are 1 and 0.01 respectively. The true and estimated outputs of nonlinear static part of the given example are compared Fig.8.6. Comparison of true and estimated system parameters along with percentage of error of linear dynamic part is shown in Table 8.5. The CPU time required for training of model structure is presented in Table 8.6. The comparative result of sum of squared errors (SSE) obtained during testing is presented in Table 8.6.

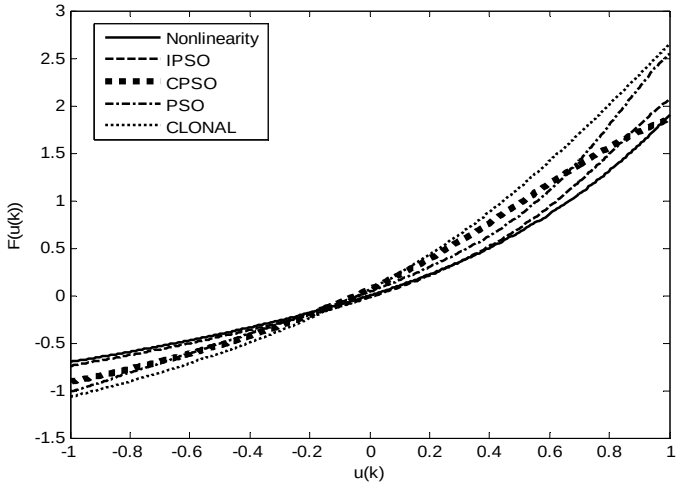


Fig. 8.6 Comparison of response at the output of nonlinear static part of the plant and the corresponding models of Example 3

Table 8.5

Comparative results of estimates of system parameters for dynamic part of the model of Example 3

Parameters	True Values	Estimated Values IPSO	CPSO	PSO	CLONAL	GA
a_1	0.9	0.898 (0.2 %)	0.900 (0.0 %)	0.841(6.5%)	1.037 (15.2%)	1.011(12.3 %)
a_2	0.15	0.146 (2.6 %)	0.069 (54.0 %)	0.150(0.0 %)	0.522 (248 %)	0.471 (214 %)
a_3	0.02	0.020 (0.0 %)	0.020 (0.0 %)	0.020(0.0 %)	0.188(940 %)	0.195 (975 %)
b_0	0.7	0.696(0.5 %)	0.750(7.14 %)	0.416(40.5 %)	0.830(18.5%)	0.846(20.8 %)
b_1	1.5	1.494(0.4 %)	1.084 (27.7 %)	0.892 (40.5%)	1.067(28.8 %)	0.992(33.8 %)

Table 8.6

Comparison of CPU time and SSE for identifying the plant of Example 3

Algorithm	During Training CPU Time (Sec.)	During Testing SSE
IPSO	53.79	3.587
CPSO	51.63	6.550
PSO	50.60	21.779
CLONAL	57.68	19.77
GA	64.71	19.96

Example 4

The Hammerstein plant used in this example is same as that of the Example3, except that a different nonlinear static part given in (8.56) is used

$$x(k) = F(u(k)) = 0.5 * \sin^3(\pi u(k)) \tag{8.56}$$

For modeling this plant the weights of the model are trained for 40 generations. In case of IPSO the values of z_i and z_f are set to 0.9 and 0.05 respectively. The remaining conditions of simulation are same as that used in example 3. The true and estimated outputs of nonlinear static part of the given example are compared in Fig. 8.7. Comparison of the estimates of the system parameters of linear dynamic part is provided in Table 8.7. Table 8.8 compares the CPU time required for training the model and sum of squared errors (SSE) obtained during testing.

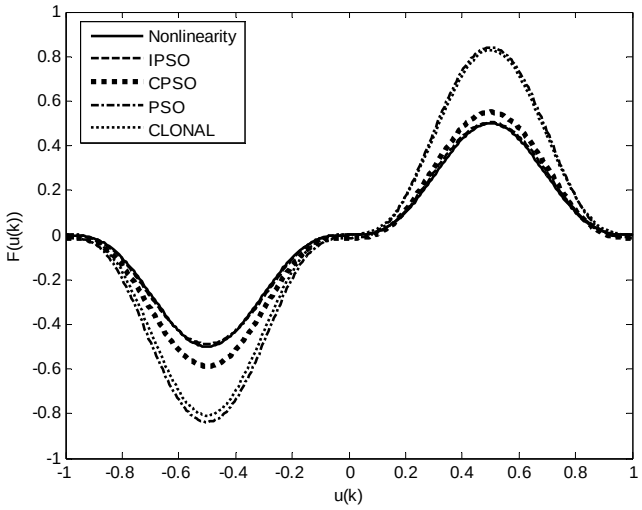


Fig. 8.7 Comparison of response at the output of nonlinear static part of the plant and the corresponding models of Example 4

Table 8.7

Comparative results of estimates of system parameters for dynamic part of the model of Example 4

Parameters	True Values	Estimated Values IPSO	CPSO	PSO	CLONAL	GA
a_1	0.9	0.890(1.1 %)	0.910 (1.1 %)	1.000(11.1 %)	0.935 (3.89 %)	0.942(4.67 %)
a_2	0.15	0.150 (0.0 %)	0.170 (13.3 %)	0.310(3.8 %)	0.240(60.0 %)	0.203(35.3%)
a_3	0.02	0.021 (5.0 %)	0.022 (10.0 %)	0.088(340 %)	0.075(275%)	0.078 (290 %)
b_0	0.7	0.690(1.4 %)	0.750(7.1 %)	0.597(14.7%)	0.462(34.0 %)	0.475(32.1%)
b_1	1.5	1.480(1.3 %)	1.471(1.93 %)	0.875 (41.6 %)	1.201(19.9 %)	1.122(25.2%)

Table 8.8

Comparison of CPU time and SSE for identifying the plant of Example 4

Algorithm	During Training CPU Time (Sec.)	During Testing SSE
IPSO	46.42	0.0016
CPSO	44.04	0.0066
PSO	44.00	0.9744
CLONAL	51.23	0.1252
GA	57.01	0.3517

8.6 Conclusion

In this Chapter two modified PSO based algorithms have been proposed by incorporating some modifications on the standard PSO algorithm and those have been used in training the weights of FLANN structure of nonlinear static part and the parameters of linear dynamic part of Hammerstein plant identification models. Both the proposed algorithms are relatively simple compared to the original PSO algorithm. But the simulation study reveals that the IPSO algorithm offers best identification performance compared to other algorithms. Out of the two algorithms proposed, the CPSO is computationally simpler but offers identification performance nearly similar to its PSO counterpart. Under identical conditions the IPSO requires more CPU time followed by PSO and CPSO. The above observations have been arrived at by comparing the SSE, the output response and the true and estimated parameters obtained from the simulation study of four benchmark examples.

References

[8.1] S.A. Billings and S. Y. Fakhouri, "Identification of Systems Containing Linear Dynamic and Static Nonlinear Elements", Automatica, vol. 18, no.1, pp. 15-26, 1982.

[8.2] F. C. Kung and D. H. Shih , " Analysis and Identification of Hammerstein Model Non-linear Delay Systems Using Block-pulse Function Expansions ", Int. J. Control, vol. 43, no. 1, pp. 139-147, 1986.

[8.3] S. Adachi and H. Murakami , "Generalized Predictive Control System Design Bused on Non-linear Identification by Using Hammerstein Model", Trans. of the Institute of Systems, Control and Information Engineers, vol. 8, no. 3, pp. 115-121, 1995.

[8.4] H. Al-Duwaish and M. N. Karim, "A New Method for the Identification of Hammerstein Model ", Automatica , vol. 33, no. 10, pp. 1871-1875, 1997.

[8.5] Y. Kobayashi, M. Oki and T. Okita , " Identification of Hammerstein Systems with Unknown Order by Neural Networks ", Trans. on the IEE Japan, vol. l20-C, no.6, pp. 871-878, 2000.

[8.6] H.X. Li, "Identification of Hammerstein models using genetic algorithms", IEE Proc. Control Theory Appl., vol. 146, no. 6, pp.499-504, 1999.

[8.7] Tomohiro Hachino, Katsuhisa Deguchi and Hitoshi Takata, "Identification of Hammerstein Model using Radial Basis Function and Genetic Algorithms", in Proc. of 5th Asian Control Conference, pp. 124-129 , 2004.

[8.8] W. Lin, Huidi Zhang and Peter X. Liu, "A New Identification Method for Hammerstein Model Based on PSO", in Proc. of IEEE International Conference on Mechatronics and automation, pp. 2184-2188, 2006 .

[8.9] W. Lin, C.G. Jiang and J.X. Qian, "The identification of Hammerstein model based on PSO with fuzzy adaptive inertia weight," J. Syst. Sci. Inf., Vol. 3, No. 2 , pp.381-391,2005.

[8.10] W. Lin and P.X. Liu, "Hammerstein model identification based on bacterial foraging," Electronics Letters, Vol. 42, No. 23, pp.1332-1333, 2006.

[8.11] R.Eberhart, J.Kennedy. "A new optimizer using particle swarm theory", in Proc. of 6th Int'l Symposium on Micro Machine and Human Science, Nagoya, Japan, pp:39-43, 1995.

[8.12] G.Panda, D. Mohanty, B.Majhi, G.Sahoo, "Identification of nonlinear systems using particle swarm optimization technique," in Proc. of IEEE Congress on Evolutionary Computation, Singapore, pp. 3253-3257, Sept. 2007.

[8.13] P. Angeline, "Evolutionary optimization versus particle swarm optimization: philosophy and performance difference", In Proc. of the Evolutionary Programming Conference, San Diago, USA, pp: 169-173, 1998.

[8.14] J. Liu, W. Xu and J. Sun , " Quantam-behaved Particle Swarm Optimization with Mutation Operator," in Proc. of IEEE Int. Conf. on Tools with Artificial Intelligence (ICTAI' 05), 2005.

[8.15] J. Liu, X. Fan and Z. Qu , " An Improved Particle Swarm Optimization with Mutation Based on Similarity," in Proc. of IEEE International Conference on Natural Computation (ICNC 07), pp.824-828,2007.

[8.16] M. Senthil Arumugam, A. Chandramohan and M.V.C. Rao, "Competitive Approaches to PSO Algorithms via New Acceleration Co-efficient variant with Mutation Operators", in Proc. of IEEE sixth Int. Conf. on Computational Intelligence and Multimedia Applications (ICCIMA'05), pp.225-230, 2005.

[8.17] V.Katari, S. Malireddi, S.K.S. Bendapudi, and G.Panda, "Adaptive Nonlinear System Identification using Comprehensive Learning PSO," in Proc. of IEEE 3rd International Symposium on Comm., Cont. and Signal Processing, pp.434-439, Mar. 2008.

[8.18] D. Dasgupta, "Advances in Artificial immune Systems," IEEE Computational Intelligence Magazine, vol. 1, issue 4, pp.40 – 49, Nov. 2006.

[8.19] D. Dasgupta and N. Attoh-Okine, "Immunity-based systems: A survey," Proc. IEEE Int. Conf. Syst., Man, Cybern., Orlando, FL, Oct. 12–15, 1997, pp. 369 –374.

[8.20] P. S. Andrews and J. Timmis, "Inspiration for the next generation of artificial immune systems, in Proc. of fourth international conference on artificial immune system" Lecture Notes in Computer Science, 3627, 126–138, 2005.

[8.21] L N de Charsto and J. V. Zuben , "Learning and Optimization using Clonal Selection Principle ," IEEE Trans on Evolutionary Computation, Special issue on Artificial Immune Systems, vol. 6,issue 3 , pp.239-251,2002.

[8.22] L N de Charsto and J. Timmis , "An Artificial Immune Network for Multimodal Function Optimization", in Proc. of IEEE Congress on Evolutionary Computation (CEC'02), vol. 1,pp.699-674,May ,Hawaii,2002.

[8.23] S. Forrest, S. A. Hofmeyr, A. Somayaji and T. A. Longstaff, "A sense of self for unix processe", in Proc. of the IEEE Symposium on Computer Security Privacy, 1996,pp. 120–128.

[8.24] S. Forrest and S. A. Hofmeyr, *Immunology as information processing : Design Principles for the Immune System and Other Distributed Autonomous Systems*, Santa Fe Institute Studies in the Sciences of Complexity, pp. 361–387, Oxford Univ. Press, 2000.

[8.25] D. W. Bradley and A. M. Tyrrell, "Immuotronics – Hardware fault tolerance inspired by the immune system", in Proc. of the third International Conference on Evolvable Systems, 2000, pp.11-20 ,1801, Springer-Verlag.

[8.26] D. W. Bradley and A. M. Tyrrell, "Immuotronics – novel finite-state-machine architectures with built-in self-test using self–nonself differentiation", IEEE Transaction on Evolutionary Computation , vol. 6, no. 3, pp. 227-238, 2002.

[8.27] Hong-Wei Ge, L. Sun, Y. C. Liang and F. Qian, „An effective PSO and AIS-based hybrid intelligent algorithm for job shop scheduling", IEEE Transaction on system, man and cybernetics A, vol. 38, no. 2, pp.358–368, 2008.

[8.28] F. Esponda, S. Forrest and P. Helman, "A formal framework for positive and negative detection", IEEE Transaction on system, man and cybernetics, vol. 34, pp.357–373, 2004.

[8.29] P Matzinger, The Danger Model: a Renewed Sense of Self. Science, 296, pp.301-304, 2002.

[8.30] J. C. Patra, R. N. Pal, B. N.Chatterji and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks" IEEE Transaction on System, Man and Cybernetics B, vol. 29, pp.254–262, 1999.

[8.31] J. C. Patra and A. C. Kot, "Nonlinear dynamic system identification using Chebyshev functional link artificial neural networks", IEEE Transaction on System, Man and Cybernetics B, vol.32, pp. 505–511, 2002.

[8.32] Tomohiro Hachino, Katsuhisa Deguchi and Hitoshi Takata, "Identification of Hammerstein Model using radial basis function and genetic algorithms", in Proc. of the 5th Asian Control Conference, Melbourne, Australia, 2004, pp. 124-129.

Chapter

9

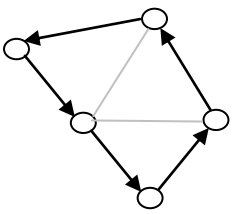
Development of Distributed Particle Swarm Optimization Algorithms for Robust Nonlinear System Identification

9.1 Introduction

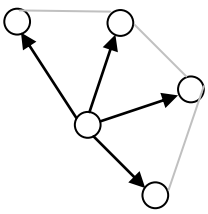
THE wireless sensor networks provide a smart interaction with the physical world and are deployed at low cost and in large numbers in remote environments. They yield autonomous and intelligent measurements, answer queries and also perform monitoring tasks. Application areas include environment monitoring, battlefield surveillance, health care, home automation and so on [9.1]. In a traditional centralized solution, the nodes in the network collect observations and send them to a central location for processing. The central processor then performs the required estimation tasks and broadcast the result back to the individual nodes. This mode of operation requires a powerful central processor, in addition to extensive amounts of communication between

the nodes and the processor. Distributed processing extracts information from data collected at different points that are distributed over a geographical area. In the distributed mode of solution, each system uses its own local data and as well as interact with its surrounding nodes to obtain global solution so that the amount of processing and communications is significantly reduced [9.2] – [9.4]. In a distributed scenario all the nodes coordinate with each other by some means and the distributed algorithm guides the system towards a promising solution without being influenced by the local information of each individual node. Though each sensor is characterized by low power constraint and limited computation and communication capabilities, potential networks can be built to perform various high level tasks with sensor collaboration [9.5] such as distributed estimation, distributed detection and target localization and tracking. Applications range from sensor networks to precision agriculture, environment monitoring, disaster relief management, smart spaces and medical applications [9.2], [9.6], [9.7]. The distributed parameter estimation has been suggested in [9.8]-[9.12] assuming that the joint distribution of sensors' observations is known and that the messages can be sent from the sensors to the fusion center without distortion.

The effectiveness of any distributed implementation depends on the modes of cooperation that are allowed among the nodes. Two such modes of cooperation available are as shown in Fig. 9.1. In the first approach the overall system parameters are estimated in a cooperative fashion by using local estimates and sharing them with their pre-identified neighbours. This mode of operation requires a cyclic pattern of collaboration among the nodes, and it tends to require the least amount of communication and power [9.3], [9.13], [9.14].



(a) Incremental



(b) diffusion

Fig. 9.1 Two modes of cooperation

The second approach employs diffusion protocols for establishing cooperation among individual nodes [9.15]-[9.17]. Each node obtains local estimates of interested parameters and share this information with their neighbours. The amount of communication in this case is higher than in an incremental case. The communications in the diffusion implementation can be reduced by allowing each node to communicate only with subset of its neighbours. Both incremental and diffusion algorithms are distributed and cooperative in nature. They are also capable of responding in real time to environmental changes.

Recently an adaptive distributed strategy is proposed based on incremental techniques for linear estimation in a cooperative fashion [9.18]. A study of distributed estimation algorithms based on diffusion protocols to implement cooperation among individual adaptive nodes is reported in [9.19]. Distributed evolutionary optimization frame work based on a swarm intelligence principle has been recently reported [9.20] for sensor networks. A different version of particle swarm optimization algorithm has been reported for a swarm of robotic applications [9.21]. An RF IC optimization methodology based on an elitist distributed particle swarm optimization algorithm is presented in [9.22].

The literature reviews reveals that the recently reported distributed algorithms have been applied only for linear estimation of interested parameters of a region. On the other hand in many practical simulations the local input and output data collected at each node of the network is nonlinearly related which means that each node functions as an individual nonlinear filter. The research work reported in this chapter is based on the following motivations :

(i) There is a need to develop novel algorithms using evolutionary computing strategy.

(ii) To assess the performance of new algorithms when used for system identification.

(iii) To evolve methodology for robust identification of plants using distributed algorithms and robust norm.

In this chapter two novel distributed PSO algorithms : incremental PSO(INPSO) and diffusion PSO (DPSO) are proposed and applied to system identification. In this method of identification minimization of mean square error is taken as the cost function. But when outliers are present in the training samples this conventional cost function does not provide satisfactory performance. Therefore to achieve robust identification performance against outliers a robust norm is introduced and used as cost function. In this chapter performance

of a new identification scheme using combination of INPSO and DPSO along with robust norm has been assesses through simulation study.

9.2 Distributed system identification

(a) INPSO based system identification

The basic concept of the incremental PSO (INPSO) algorithm is as follows :
Each node performs its own calculation using its local data, updates its particles' positions, velocities, personal best (pbest) positions and transfer the global best position to its next neighbour. The adjacent node updates its own particles' positions, velocities and pbests using its local data and the gbest vector received from the previous node and transmit the new updated gbest to its neighbour. This process continues for several cycles through the network until the desired solution is obtained. Fig. 9.2 shows the schematic representation of an INPSO based identification scheme. In this scheme it is assumed that $N \quad (1 \le n \le N)$ sensor nodes present in a small region, participate in the identification task using their local measurements $\{X_n, D_n\}$. In this case $X_n = [x_{n1}\ x_{n2}.....x_{nM}]^T$ and $D_n = [d_{n1}\ d_{n2}.....d_{nM}]^T$ represent the input and desired samples of n th node. Each desired sample is contaminated with white Gaussian noise $v_m(n)$. At each n th node the swarm consists of K number of particles. In any l th $(1 \le l \le L)$ search, the position vector, the best position vector and the velocity vector of k th particle of first node is represented by $P_{1k}(l), B_{1k}(l)$ and $V_{1k}(l)$ respectively. During a specific search, a pair of input-output samples $\{x_m, d_m\}$ produces an error and hence a total of M errors are generated for a particle. The mean square error, $E_{1k}(l)$ of the k th particle at l th search is thus computed and is used as the fitness or cost function. The initial position vector of any k th particle is taken to be its initial personal best position vector. From the set of $B_{1k}(l)$, $1 \le k \le K$, the best position vector corresponding to the least $E_{1b}(l)$ is chosen as the global best position vector. The values of $P_{1k}(l), B_{1k}(l), V_{1k}(l)$ associated with k th

population of first node and the global best position vector $G_N(l)$ of N th node are used to update the velocity vector and then the position vector. These update equations are

$$V_{1k}(l+1) = w_l\, V_{1k}(l) * c_{1k} * rand_{1k}(l) * [B_{1k}(l) - P_{1k}(l)] + c_{2k} * rand_{2k}(l) * [G_1(l) - P_{1k}(l)] \quad (9.1)$$

$$P_{1k}(l+1) = P_{1k}(l) + V_{1k}(l+1) \quad (9.2)$$

where $P_{1k}(l), B_{1k}(l)$ are position and local best position vectors at l th search of k th particle of node 1. $G_1(l)$ represents the global best position vector at l th search of node 1. $rand_{1k}(l)$ and $rand_{2k}(l)$ are random numbers in the range [0, 1]. c_1 and c_2 represent the acceleration coefficients and w is the inertia weight which balances the global and local searches. The inertia weight at l th search is given by

$$w(l) = w_0 - \frac{(w_0 - w_1) * l}{L} \quad (9.3)$$

where

l = search number

L = total number of searches

$w_0 = 0.9$ and $w_1 = 0.4$

The dotted portion of Fig. 9.2 indicated by A_1 corresponds to node 1 of the incremental sensor network. The contents of block A_2, A_{N-1} and A_N of Fig. 9.2 are identical to that of A_1 except that the initialized values of the positions and velocities of the particles are different. The updated values are used to obtain the new personal best and global best position vectors required for next search. After completion of l th search at the k th node, the global position vector $G_n(l)$ is evaluated and communicated to $(k+1)$ th neighbour. This process is repeated until the average MSE of the cluster of nodes reduces to the lowest possible value. Under this situation, the final global best position vector gives the estimate of the weight vector of the model. On the other hand identification of nonlinear system is achieved when the response of the model matches with the plant output.

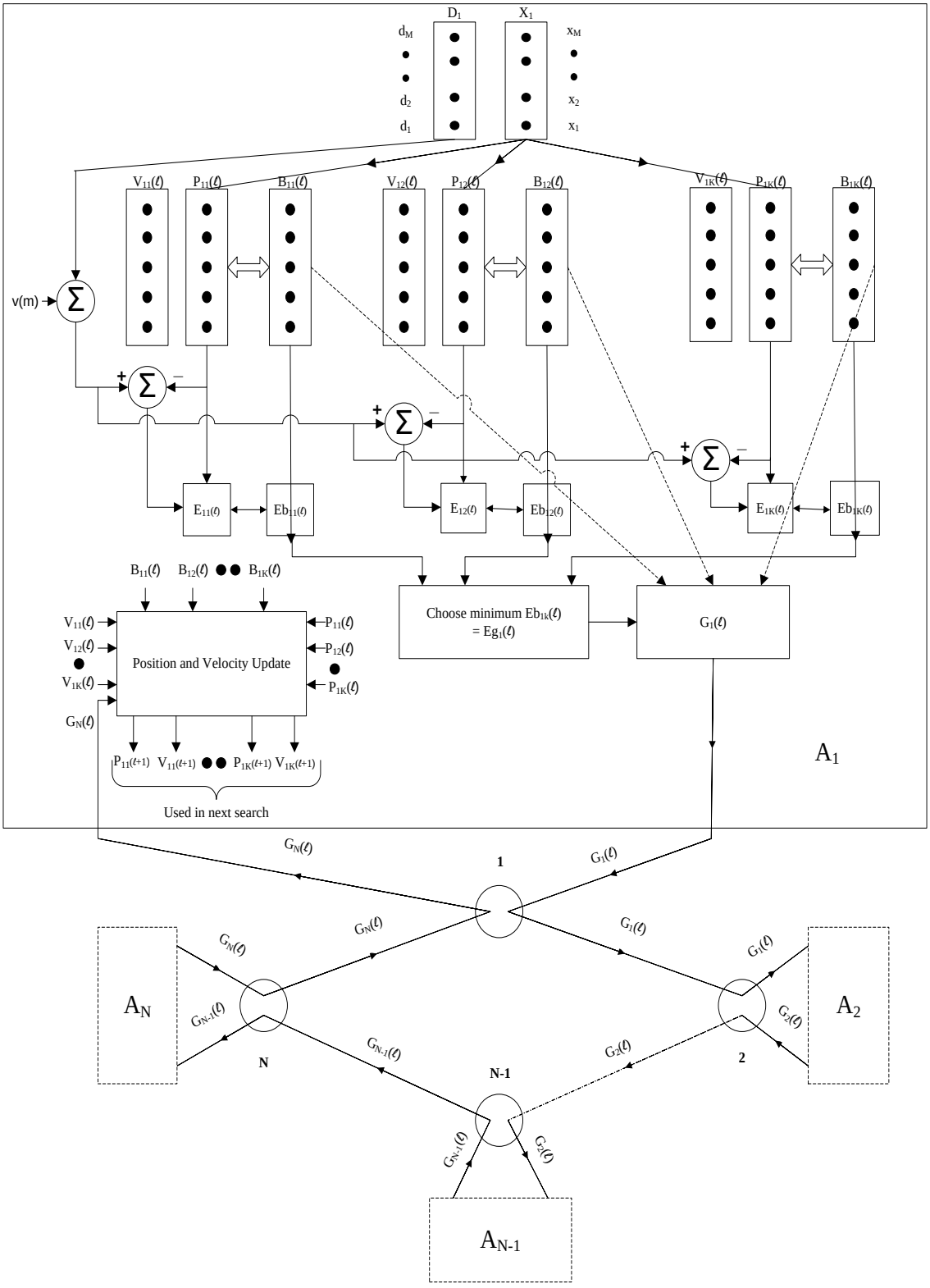


Fig. 9.2 INPSO based nonlinear identification scheme

(b) DPSO based system identification

The diffusion based distributed mechanism using LMS is reported in [8.9]. Diffusion PSO (DPSO) algorithm uses similar cooperation strategy. The main objective in this case is to develop an adaptive distributed procedure that approximates the solution and delivers a good estimate to every node in the network. In DPSO each node updates its particles' positions, velocities and pbest using its local data and transmits the gbest vector to all other neighbouring nodes. Every node then finds out the best gbest and employs this gbest and its local available data for computing its new positions, pbest and gbest. Such an update strategy reduces the amount of information that is communicated among the particles and thus increases the stability of the algorithm. The information exchange mechanism of DPSO is shown in Fig. 9.3.

Like in the case of INPSO, the global best position, G_n at all nodes are evaluated using local data. Similarly the position and velocities of all particles at nodes are simultaneously evaluated. The computed global best information G_n is exchanged between all participating nodes. Then each node locally compares and selects the best of global best positions (G_{bn}) and use it for updating the velocity and position of particles of that node. The velocity and position update equations remain same as given in (9.1) and (9.2). The above stated procedure is repeated until the average MSE of all nodes attains the lowest possible value.

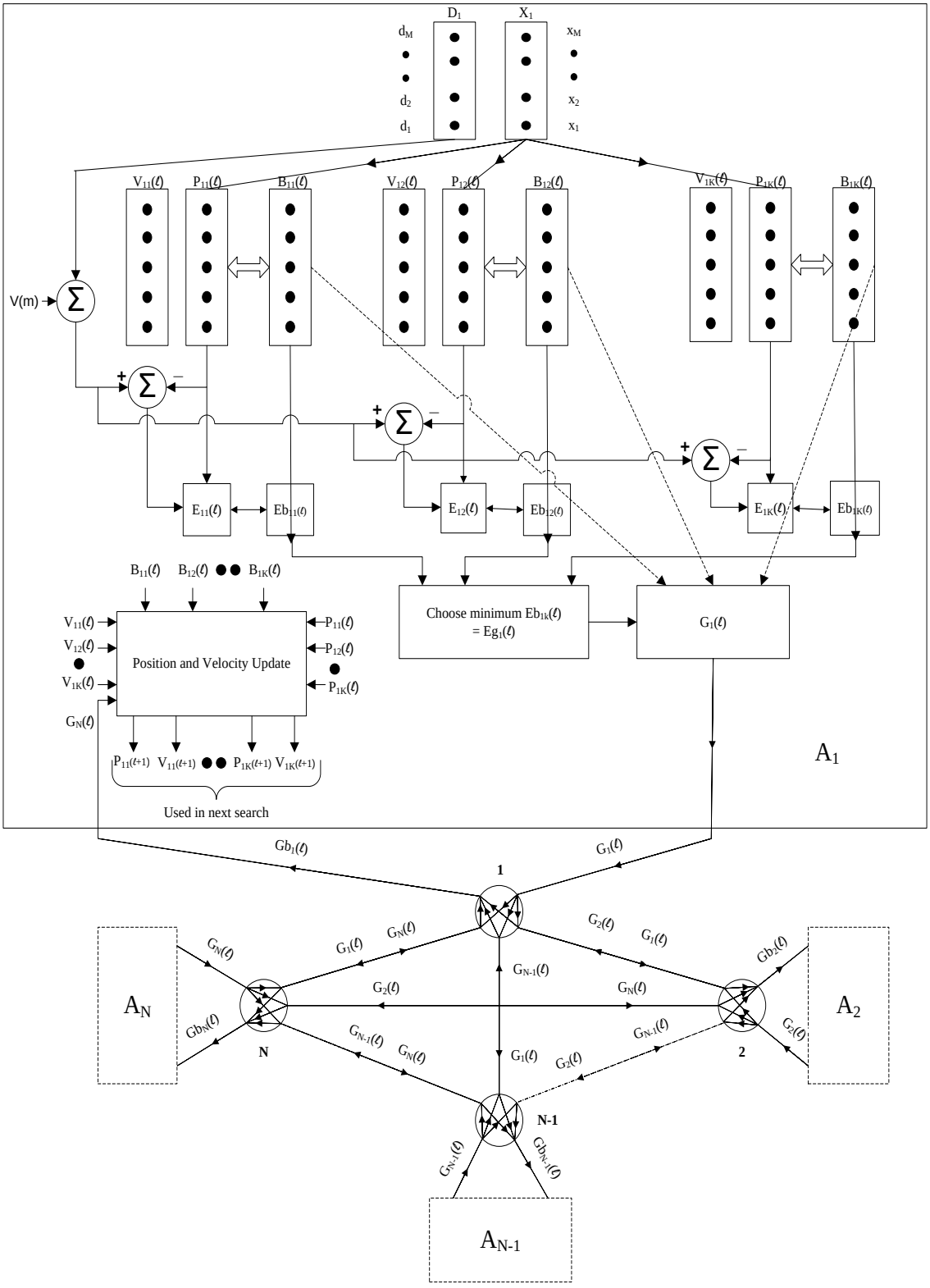


Fig. 9.3 DPSO based nonlinear identification scheme

9.3 Distributed robust identification of plants

When the measured data contains outliers of different densities and strengths then the conventional learning algorithms based on squared error minimization provide poor identification performance. This is because of improper training of the identification model and hence the scheme is not robust. To improve the identification performance under such adverse conditions the learning strategy needs improvement. Attempt has been made to fulfill this objective by selecting and employing robust norm from the statistics literature. One such norm is Wilcoxon norm [9.27] which has been proven to be a robust norm against outliers. The detail of this norm is available in Section 6.4 of Chapter 6.

The distributed strategy of training used for identification using INPSO and DPSO algorithms outlined in the previous sections are almost identical. The only exception in this case is a new cost function which is the Wilcoxon norm evaluated from the error vector. This norm of errors is minimized using INPSO and DPSO algorithms as suggested in the previous section. The performance of these two algorithms in identification of plant is evaluated through simulation study and is presented in subsequent section.

9.4 Stepwise distributed PSO algorithms

The objective of an adaptive identification algorithm is to change the coefficients of the model iteratively so that the squared error, $e^2(k)$ or any other robust norm (Wilcoxon norm) is minimized and subsequently reduced to a best possible minimum. In this case the architecture of the model is an adaptive linear combiner and the learning algorithm used to change the coefficient values is either INPSO or DPSO. The steps involved in these two algorithms are outlined as follows :

1 M $(M \geq 100)$ number of input signal samples uniformly distributed between -0.5 to +0.5 are generated.

2 Ten of these input samples are assigned to each node. In this way all 100 samples are assigned to all 10 nodes.

3 Each ten input samples is passed to the nonlinear model of a corresponding node and measurement noise of known strength is added to the output. The resultant signal acts as the desired output at that node. This process is repeated at all nodes

using their corresponding models and the estimated output in each is obtained. This process is repeated for all nodes.

4 By comparing the output sample with the corresponding estimated output of the model the error signal is obtained.

5 The mean square error (MSE) defined in (9.4)/Wilcoxon norm defined in (6.12) of k th particle is determined

$$MSE(k) = \frac{\sum_{m=1}^{M} e_m^2}{M} \tag{9.4}$$

This is repeated for all particles.

6 The MSE/Wilcoxon norm is minimized by using INPSO and DPSO based techniques using the procedure outlined in Figs. 9.2 and 9.3 respectively.

7 The velocity and position of each particle is updated using (9.1) and (9.2).

8 In case of INPSO, the gbest is transmitted to the next node, but in case of DPSO, the same value is transmitted to all other neighbouring nodes. This process is repeated for each of ten nodes.

9 In each iteration, the average MSE/average Wilcocon norm (expressed in dB) is stored and is plotted against iteration(cycle) to show the learning characteristics of the distributed PSO algorithm.

10 When the MSE/Wilcoxon norm reaches the pre-specified level, the optimization process is stopped.

11 At this stage all the particles attain almost the same positions which represent the estimated coefficient vector of the overall system.

To validate and compare the performance of proposed INPSO and DPSO algorithms the conventional PSO algorithm is also simulated to obtain the estimated parameters. To achieve this all the data collected at different nodes are accumulated at one node and are used in the identification task [9.23].

9.5 Simulation study

Identification of standard linear and nonlinear plants is carried out under distributed environment using newly developed INPSO and DPSO algorithms. Zero mean white

Gaussian noise is added to the plant output to generate the training signal. To facilitate comparison estimation of parameters is also carried out using centrally available measured data and PSO algorithm for training the model.

A. Identification of linear plants

Three non-minimum phase all-zero systems [9.24, 9.25] used in this simulation are

$1.\ x(n) = w(n) + 0.9w(n-1) + 0.385w(n-2) - 0.771w(n-3)$

$2.\ x(n) = w(n) - 0.8w(n-1) + 1.52w(n-2) - 0.64w(n-3) + 0.99w(n-4)$

$3.\ x(n) = w(n) - 2.33w(n-1) + 0.75w(n-2) + 0.5w(n-3) + 0.3w(n-4) - 1.4w(n-5)$

The zeros of these systems are located at $-0.75 \pm j\,0.85$ and 0.6, $0.6 \pm j\,0.8602$ and $-0.2 \pm j0.9274$ and $1.851, -0.587 \pm j\,0.563$ and $0.827 \pm j\,0.678$ respectively. The output of the plant is added with white Gaussian noise of strength -30dB to produce the output of the known plants. Uniform random signal within the range of -0.5 to +0.5 is generated and used as input signal. This is passed through both the plant and adaptive model simultaneously. Table 9.1 shows the number of nodes and number of input samples used in the simulation study. It is observed that the product of number of nodes and number of input samples are kept constant in all cases. The parameters estimated using distributed PSO algorithm are listed in Table 9.2. This Table also shows the percentage deviation of coefficients, the MSE during testing and training time in seconds obtained by three different methods.

Table 9.1
Comparison of simulation parameters used in INPSO, DPSO and PSO based models

Example no.	Parameters used in simulation			
Ex-1		INPSO	DPSO	PSO
	No. of nodes	10	10	1
	No. of input samples at each node	50	50	500
	No. of particles at each node	10	10	100
Ex-2	No. of nodes	10	10	1
	No. of input samples at each node	100	100	1000
	No. of particles at each node	100	100	1000
Ex-3	No. of nodes	10	10	1
	No. of input samples at each node	100	100	1000
	No. of particles at each node	60	60	600

Table 9.2
Comparison of estimated parameters using INPSO, DPSO and PSO techniques

Example no.	Actual Parameters	Estimated parameters					
		INPSO	% of Deviation	DPSO	% of Deviation	PSO	% of Deviation
Ex -1	1 0.9 0.385 -0.771	0.999 0.8988 0.3842 -0.7682	0.0100 0.1333 0.2078 0.3632	1.0000 0.9007 0.3906 -.7735	0 0.0778 1.4545 0.3243	0.999 0.9015 0.3840 -.7714	0.1000 0.1667 0.2597 0.0519
MSE during testing		9.13×10^{-4}		8.84×10^{-4}		9.16×10^{-4}	
Execution time in second		0.1260		0.1143		1.0888	
Ex-2	1 -0.80 1.52 -0.64 0.99	0.9882 -0.8021 1.4948 -0.6466 0.9763	1.1800 0.2625 1.6600 1.0312 1.3838	0.9998 -0.7965 1.4983 -0.6369 0.9855	0.0200 0.4375 1.4276 0.4844 0.4545	0.9858 -0.8007 1.4991 -0.6484 0.9869	1.4200 0.0875 1.3750 1.3125 0.3131
MSE during testing		0.0050		0.0037		0.0013	
Execution time in second		2.1755		2.1617		21.6401	
Ex-3	1 -2.33 0.75 0.5 0.3 -1.40	0.9923 -2.3216 0.7381 0.5272 0.3207 -1.2800	0.7700 0.3605 1.5867 5.4400 6.9000 8.5700	0.9986 -2.3779 0.6928 0.5094 0.2563 -1.2929	0.1400 2.0558 7.6267 1.8800 14.5667 7.6500	0.9994 -2.1757 0.9921 0.5232 0.3154 -1.2543	0.0600 6.6223 32.2800 4.6400 5.1333 10.410
MSE during testing		0.0476		0.0561		0.1798	
Execution time in second		3.2648		3.2563		32.5122	

Observation of results indicates that both distributed methods provide excellent identification performance which is comparable or even better. Further the training time of the new methods is much less compared to that of conventional PSO based method.

B. Identification of nonlinear plants

The new algorithms are also used to estimate nonlinear parameters. Four nonlinear plants are simulated using combinations of two linear plants cascaded with two different nonlinearities [9.26].

Linear plants

4. $H(z) = 0.3040 + 0.9030z^{-1} + 0.3040z^{-2}$ and

5. $H(z) = 0.2600 + 0.9300z^{-1} + 0.2600z^{-2}$

Two different nonlinearities used to make nonlinear plants are

NL1 : $b(k) = \tanh\{a(k)\}$

NL2 : $b(k) = a(k) + 0.2a^2(k) - 0.1a^3(k)$

where $a(k)$ is the output of the linear part of the node $H(z)$ and $b(k)$ is the output of the non-linear part of the node. The output of the system is added with white Gaussian noise of different strengths of *–20dB* and *–30dB*. The convergence characteristics of INPSO, DPSO and conventional PSO obtained from simulation study for the nonlinear estimation problem of Examples 4 and 5 are shown in Figs.9.4 (a)-(h) for -30dB and -20dB additive noise. The convergence characteristics shown in Figs. 9.4(a)-(h) also indicate that the new distributed approaches converge to noise floor levels below that achieved by the PSO method. However the convergence performance of DPSO and INPSO is observed to be almost identical.

The output responses obtained by these methods are compared during test phase and are shown in Figs.9.5 (a) and (b). The comparison shows excellent agreement between the actual and estimated responses obtained by new methods. In addition the response achieved by the new methods is also comparable with that of conventional PSO based method. Table.9.3 provides the CPU time taken by the new algorithms during training phase when they are implemented under identical conditions. It also lists the sum of squared error (SSE) obtained from actual and estimated responses for different examples. Observation of this Table indicates that at both noise levels the CPU time taken by INPSO and DPSO based algorithms is less in comparison to that obtained from PSO model. Therefore, considering all counts the INPSO and DPSO approaches are observed to be better distributed candidates for parameter estimation or response matching of complex plants.

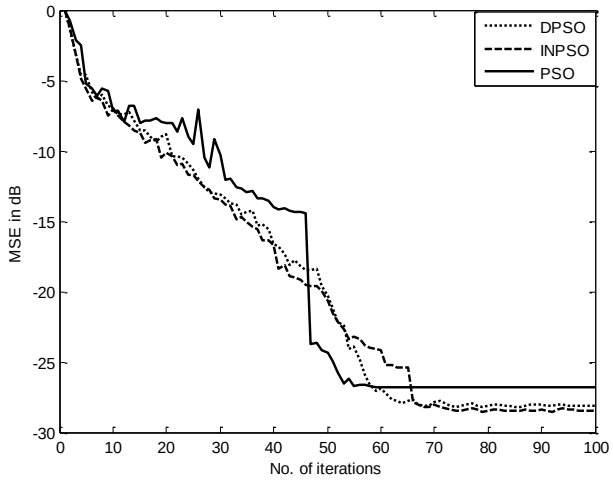


Fig 9.4 (a) Convergence of Ex-4 with NL1 at -30dB

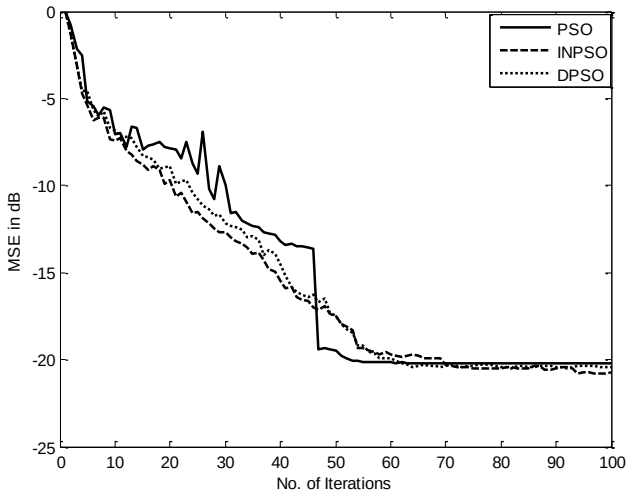


Fig 9.4 (b) Convergence of Ex-4 with NL1 at -20dB

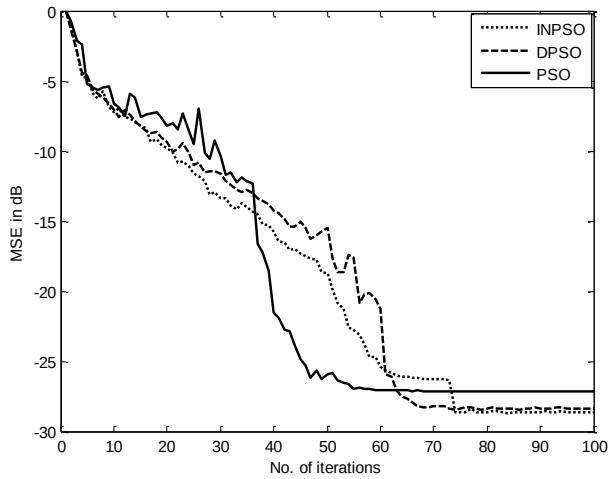


Fig. 9.4 (c) Convergence of Ex-5 with NL1 at -30dB

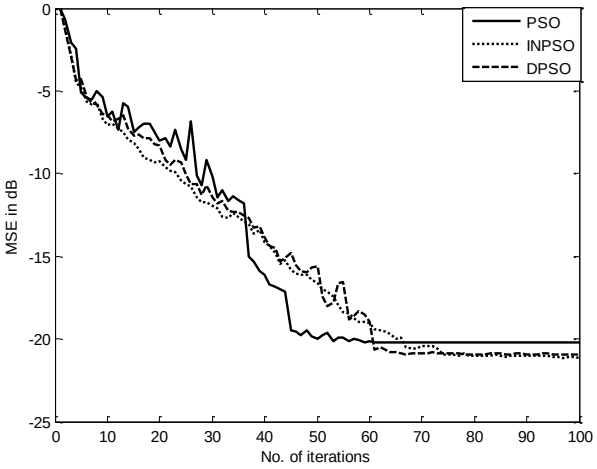


Fig. 9.4 (d) Convergence of Ex-5 with NL1 at -20dB

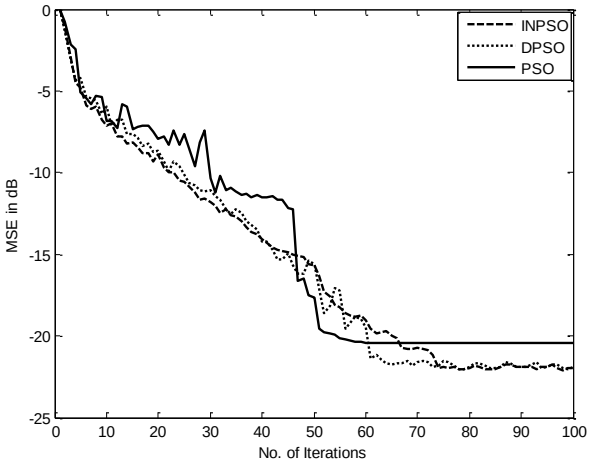


Fig. 9.4 (e) Convergence of Ex-4 with NL2 at -30dB

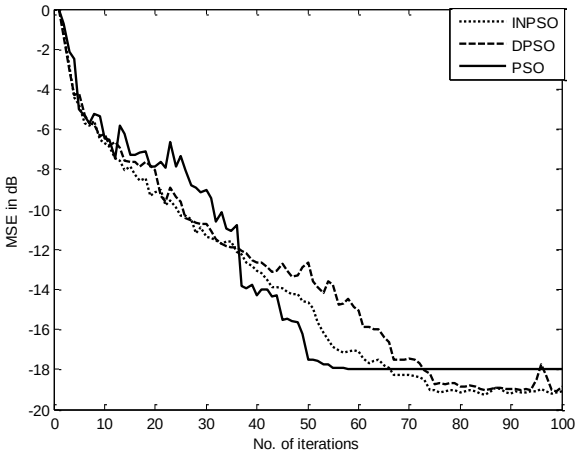


Fig. 9.4 (f) Convergence of Ex-4 with NL2 at -20dB

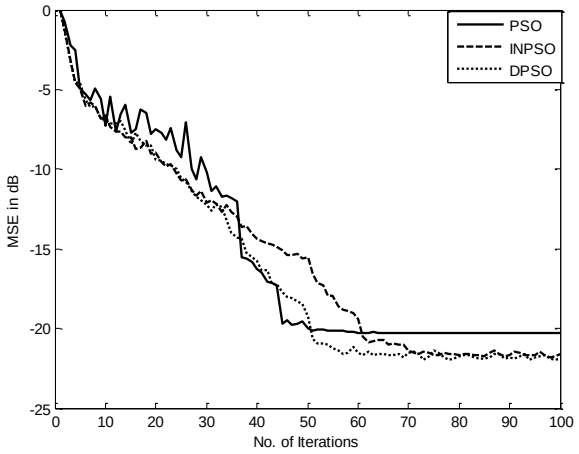


Fig. 9.4 (g) Convergence of Ex-5 with NL2 at -30dB

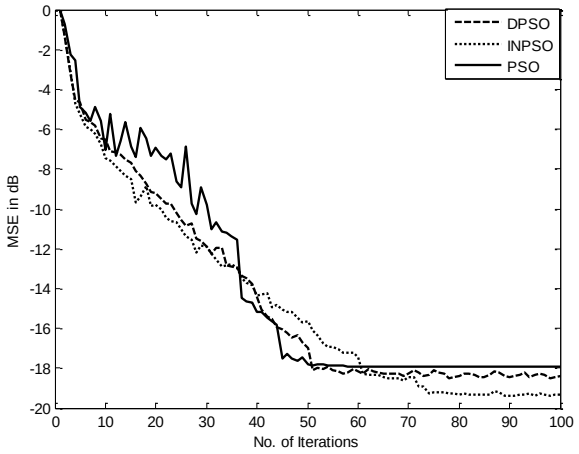


Fig. 9.4 (h) Convergence of Ex-5 with NL2 at -20dB

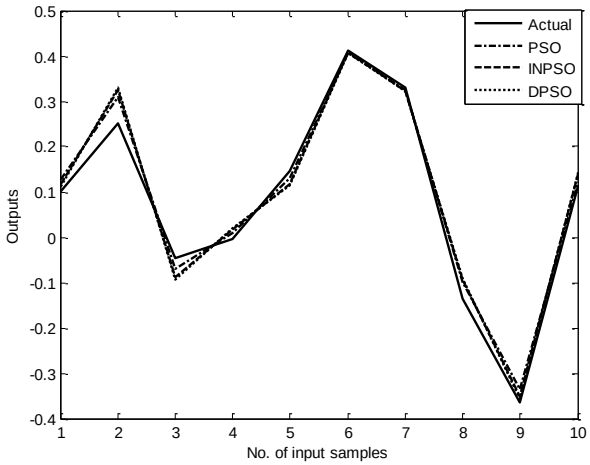


Fig. 9.5 (a) Response matching of Ex-4 with NL1 at -20dB

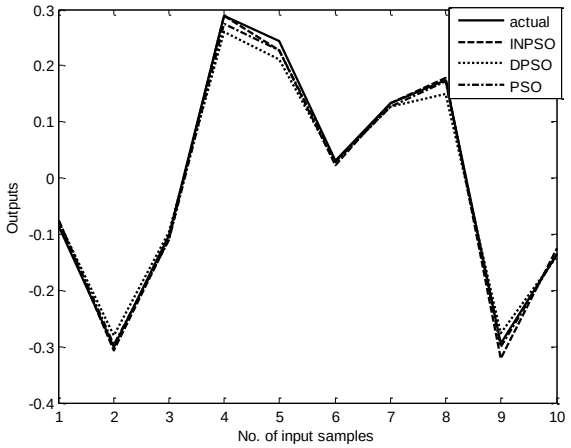


Fig. 9.5 (b) Response matching of Ex-5 with NL2 at -30dB

Table 9.3
Comparison of CPU time and sum of squared error obtained using PSO, INPSO and DPSO

Examples	CPU Time in second during training			Sum of squared error (SSE) during testing		
	PSO	INPSO	DPSO	PSO	INPSO	DPSO
Ex-4 with NL1 at -30dB	.0200	.0141	.0133	.0009	.0013	.0007
Ex-4 with NL1 at -20dB	.0240	.0156	.0135	.0077	.0117	.0134
Ex-4 with NL2 at -30dB	.0216	.0156	.0136	.0026	.0022	.0074
Ex-4 with NL2 at -20dB	.0195	.0159	.0138	.0067	.0120	.0127
Ex-5 with NL1 at -30dB	.0203	.0156	.0136	.0008	.0012	.0006
Ex-5 with NL1 at -20dB	.0206	.0154	.0135	.0076	.0117	.0068
Ex-5 with NL2 at -30dB	.0219	.0159	.0138	.0027	.0034	.0079
Ex-5 with NL2 at -20dB	.0208	.0159	.0141	.0064	.0105	.0129

C. Robust distributed identification of nonlinear plants

Here the training signal is contaminated with outliers at locations ranging from 10% to 50%. The magnitudes of such disturbance are varied between -5 to +5, -10 to +10 and -20 to +20. The Wilcoxon norm of errors is used to train the model using INPSO and DPSO algorithms. For comparison purpose the squared error norm based models are also simulated. In all cases the SSE is obtained as performance measure to compare the performance between the two methods. Simulation results presented in Tables 9.4 – 9.7 clearly indicate that the Wilcoxon norm based distributed INSPO and DPSO algorithms show least SSE in all cases compared to their squared error counterpart. This is true for all variations of density and strength of outliers studied. In presence of outliers both the robust distributed PSOs are observed to provide almost identical identification performance.

9.6 Conclusion

The chapter presents two distributed PSO algorithms : INPSO and DPSO and use them for identification of nonlinear plants using locally measured data. In all cases it is observed that the two algorithms provide almost similar performance. Further it is observed that the two algorithms do not provide satisfactory identification performance when outliers are present in the training signal. To enhance the robustness of these two algorithm Wilcoxon norm based training is incorporated in this chapter. The results of identification using such training scheme indicate that with high density outliers up to 50% and with strength of outliers as high as -20 to 20, the proposed distributed approach shows more robust performance in identification of nonlinear plants.

Table 9.4

Comparison of sum of squared error (SSE) during testing for Ex-4 with nonlinearity NL1

Percentage of outliers	INPSO using		DPSO using	
	Wilcoxon norm	Error square	Wilcoxon norm	Error square
Outlier within the range of (-5 to +5)				
0%	.0009	.0009	.0009	.0010
10%	.0012	.1546	.0013	.2379
20%	.0011	.4424	.0009	.2317
30%	.0024	.2639	.0009	.1038
40%	.0022	.2170	.0023	1.8967
50%	.0049	1.4475	.0065	1.1638
Outlier within the range of (-10 to +10)				
10%	.0011	.9845	.0013	.8314
20%	.0013	.4424	.0009	.5018
30%	.0014	.2230	.0009	.1125
40%	.0029	.7623	.0030	2.4375
50%	.0032	2.5405	.0114	3.0345
Outlier within the range of (-20 to +20)				
10%	.0011	1.7893	.0013	1.5053
20%	.0011	.8412	.0009	.6650
30%	.0014	1.4450	.0009	.0474
40%	.0025	.9239	.0038	2.4375
50%	.0066	2.3882	.0114	2.9731

Table 9.5

Comparison of sum of squared error (SSE) during testing for Ex-4 with nonlinearity NL2

Percentage of outliers	INPSO using		DPSO using	
	Wilcoxon norm	Error square	Wilcoxon norm	Error square
Outlier within the range of (-5 to +5)				
0%	.0053	.0055	.0048	.0049
10%	.0068	.2426	.0063	.2423
20%	.0060	.4347	.0052	.2234
30%	.0048	.2520	.0056	.1087
40%	.0067	.2003	.0102	1.9191
50%	.0164	1.8002	.0206	1.2163
Outlier within the range of (-10 to +10)				
10%	.0061	1.0322	.0063	.8194
20%	.0054	.4347	.0052	.4813
30%	.0057	.1941	.0056	.0183
40%	.0086	.8069	.0112	2.4849
50%	.0134	2.2508	.0222	3.2130
Outlier within the range of (-20 to +20)				
10%	.0061	2.12	.0063	1.4678
20%	.0062	.8495	.0052	.6416
30%	.0051	1.3547	.0056	.1548
40%	.0087	.9210	.0136	2.4849
50%	.0160	2.5370	.0222	3.1524

Table 9.6

Comparison of sum of squared error (SSE) during testing for Ex-5 with nonlinearity NL1

Percentage of outliers	INPSO using		DPSO using	
	Wilcoxon norm	Error square	Wilcoxon norm	Error square
Outlier within the range of (-5 to +5)				
0%	.0008	.0067	.0008	.0071
10%	.0013	.2434	.0011	.2937
20%	.0012	.4424	.0007	.2778
30%	.0019	.2639	.0008	.1022
40%	.0025	.2841	.0022	1.8401
50%	.0050	1.8163	.0068	1.1896
Outlier within the range of (-10 to +10)				
10%	.0012	.9888	.0011	.8712
20%	.0012	.4424	.0008	.5170
30%	.0014	.1728	.0008	.1550
40%	.0025	.9108	.0032	2.4375
50%	.0042	1.8111	.0105	3.0536
Outlier within the range of (-20 to +20)				
10%	.0012	2.1028	.0011	1.5606
20%	.0011	.8412	.0008	.6981
30%	.0013	1.3217	.0008	.1296
40%	.0015	.9397	.0036	2.4375
50%	.0039	2.3882	.0102	2.9796

Table 9.7

Comparison of sum of squared error (SSE) during testing for Ex-5 with nonlinearity NL2

Percentage of outliers	INPSO using		DPSO using	
	Wilcoxon norm	Error square	Wilcoxon norm	Error square
Outlier within the range of (-5 to +5)				
0%	.0051	.0083	.0043	.0100
10%	.0057	.3227	.0058	.2994
20%	.0053	.4347	.0047	.2693
30%	.0048	.2520	.0052	.1065
40%	.0060	.2026	.0097	1.8385
50%	.0149	2.2513	.0236	1.2425
Outlier within the range of (-10 to +10)				
10%	.0073	1.0266	.0058	.8571
20%	.0049	.4347	.0047	.4977
30%	.0049	.1941	.0052	.1030
40%	.0060	1.0471	.0112	2.4849
50%	.0106	1.9260	.0243	3.2359
Outlier within the range of (-20 to +20)				
10%	.0073	2.2317	.0058	1.5579
20%	.0051	.8495	.0047	.6686
30%	.0050	1.3819	.0052	.1354
40%	.0071	.9210	.0130	2.4849
50%	.0171	2.5370	.0257	3.1592

References

[9.1] I. F. Akyildiz, W. Su, Y. Sankarsubramaniam and E. Cayirci, "Wireless sensor networks: A survey", Computer Netw., vol. 38, pp. 393-422, March 2002.

[9.2] D. Estrin, G. Pottie and M. Srivastava, "Intrumenting the worlds with wireless sensor networks", in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP), Salt Lake City, UT, May 2001, pp. 2033-2036.

[9.3] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization", IEEE J. Sel. Areas Commun., vol. 23, no. 4, pp. 798-808, April 2005.

[9.4] M. Wax and T. Kailath, "Decentralized processing in sensor arrays", IEEE Trans. Acoust, Speech, Signal Processing, vol. ASSP-33, no. 4, pp. 1123-1129, October 1985.

[9.5] S. Kumar, F. Zhao and D. Shepherd, "Special issue on collaborative information processing", IEEE Signal Processing Mag., vol. 19, no. 2, pp. 13-14, March 2002.

[9.6] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A survey on sensor networks", IEEE Communication Magazine, vol. 40, no. 8, pp. 102-114, August 2002.

[9.7] D. Culler, D. Estrin and M. Srivastava, "Overview of sensor networks", Computer, vol. 37, no. 8, pp. 41-49, August 2004.

[9.8] D. A. Castanon and D. Teneketzis, "Distributed estimation algorithms for nonlinear systems", IEEE Trans. Autom. Control, vol. AC-30, pp. 418-425, 1985.

[9.9] A. S. Willsky, M. Bello, D. A. Castanon, B. C. Levy and G. Verghese, "Combining and updating of local estimates and regional maps along sets of one-dimensional tracks", IEEE Trans. Autom. Control, vol. AC-27, pp. 799-813, 1982.

[9.10] Z. Chair and P. K. Varshney, "Distributed Bayesian hypothesis testing with distributed data fusion", IEEE Trans. Syst. Man, Cybern., vol. 18, pp. 695-699, 1988.

[9.11] J. L. Speyer, "Computation and transmission requirements for a decentralized linear-quadratic Gaussian control system", IEEE Trans. Autom. Control, vol. AC-24, pp. 266-269, 1979.

[9.12] C. Y. Chong, "Hierarchical estimation", in 2nd MIT/ONR workshop on C3, Monterey, CA, Jul. 1979.

[9.13] M. G. Rabbat and R. D. Nowak, "decentralized source localization and tracking", in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP), Montreal, QC, Canada, May 2004, vol. 3, pp. 921-924.

[9.14] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: Formulation and performance analysis", in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP), Toulouse, France, May 2006, vol. 3, pp. 584-587.

[9.15] L. Xiao, S. Boyd and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus" in Proc. 4th Int. Symp. Information Processing in Sensor Networks, Los Angeles, CA, April 2005, pp. 63-70.

[9.16] L. Xiao, S. Boyd and S. Lall, "A space-time diffusion scheme for peer-to-peer least-squares estimation", in Proc. 5th Int. Symp. Information Processing in Sensor Networks, Nashville, TN, april 2006.

[9.17] C. G. Lopes and A. H. Sayed, "Diffusion least mean squares over adaptive networks", in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP), Honolulu, HI, April 2007, pp. 917-920.

[9.18] Cassio G. Lopes and Ali H. Sayed, "Incremental adaptive strategies over distributed networks", IEEE Trans. on Signal Processing, vo. 55, no. 8, pp. 4064-4077, August 2007.

[9.19] Cassio G. Lopes and Ali H. Sayed, "Diffusion least mean squares over adaptive networks: Formulation and performance analysis", IEEE Trans. on Signal Processing, vol. 56, no. 7, pp. 3122-3136, July 2008.

[9.20] Bo Wang and Zhihai He, "Distributed optimization over wireless sensor networks using swarm intelligence", in IEEE Int. Symposium on Circuits & Systems, 2007, pp. 2502-2505.

[9.21] James M. Hereford, "A distributed particle swarm optimization algorithm for swarm robotic applications", in IEEE Congress on Evolutionary Computation, Canada, 2006, pp. 1678-1685.

[9.22] Min Chu and David J. Allstot, "An elitist distributed particle swarm algorithm for RF IC optimization", in Asia and South Pacific Design Automation Conference, 2005, vol. 2, pp. 671-674.

[9.23] G. Panda, D. Mohanty, Babita Majhi and G. Sahoo, "Identification of Nonlinear Systems using Particle Swarm Optimization Technique", Proc. of IEEE Congress on Evolutionary Computation(CEC-2007), Singapore, 25-28, September, 2007, pp.3253-3257.

[9.24] Xian-Da Zhang and Yuan-Sheng, "FIR system identification using HOS alone", IEEE Trans. on Signal Processing, vol 42, no. 10, pp. 2854-2858, October 1994.

[9.25] Wei Li and Wan-Chi Siu, "New approaches without post processing to FIR system identification using selected order cumulants", IEEE Trans. on Signal Processing, vol. 48, no. 4, pp. 1144-1153, April 2000.

[9.26] J. C. Patra, R. N. Pal, R. Baliarsingh and G. Panda, "Nonlinear channel equalization for QAM signal constellation using Artificial Neural Network, IEEE Trans. On Systems, Man and Cybernetics – Part B: vol. 29, No. 2, pp.262-272, April 1999.

[9.27] Jer-Guang Hsieh, Yih-Lon Lin and Jyh-Horng Jeng, "Preliminary study on Wilcoxon learning machines", IEEE Trans. on neural networks, vol. 19, no. 2, pp. 201-211, Feb. 2008.

Chapter

10

Conclusion and Scope for Further Work

10.1 Conclusion

IN this chapter the conclusion of the overall thesis is presented and some of the future research problems which may be attempted by interested readers are outlined. The thesis has investigated on two key problems : efficient direct modeling or system identification of complex and noisy plants like Hammerstein, dynamic SISO and MIMO and inverse modeling (normal and robust) of nonlinear channels. The novelty of the present work is the introduction of bio-inspired techniques to direct and inverse modeling problems. These techniques have been essentially used for training the weights of the model. The bio-inspired techniques used are (i) PSO (ii) modified PSO such as CLPSO, CPSO, IPSO and (iii) BFO. In **Chapter 3** a new cascaded FLANN (CFLANN) structure using a novel learning algorithm is employed for identification of nonlinear dynamic plants. It is shown through exhaustive simulation that the new model offers least computation compared to MLANN and FLANN models. In all cases studied, the proposed CFLANN has produced improved response matching and least sum of squared errors between the actual and estimated outputs [10.3].

Identification of standard IIR plants has been carried out in **Chapter 4** using an improved PSO (CLPSO) based algorithm. In terms of convergence behaviour, execution time and product of population size and input samples, it is observed that the new IIR identification scheme offers improved performance compared to RLMS, GA and PSO based methods. Further it is observed that the proposed method shows better convergence characteristics than the GA and PSO based methods [10.5] when the reduced order models are used. This shows that the new method is capable of offering better optimal solution compared to GA and PSO based identification schemes.

In **Chapter 5**, two new bio-inspired techniques called PSO and BFO are introduced to develop efficient identification models for nonlinear dynamic SISO and MIMO plants. In all cases the structure used is essentially a low complexity FLANN. The population based PSO and BFO techniques provide improved learning of the models compared to that provided by gradient descent algorithm. Further, the computation time offered by the new methods is less than the existing BP trained models. It is further observed that the identification performance of BFO and PSO based models is almost similar [10.1, 10.4, 10.6, 10.9, 10.14] but BFO based model is computationally faster than the PSO based model during training.

If squared error cost function is used, the training data contaminated with outliers degrades the identification performance. In **Chapter 6**, three robust norms are introduces as the cost functions and PSO based training to design robust identification models has been suggested. These new models produce improved identification of complex plants even when 50% outliers in the training samples are present. Out of the three robust norms used it is shown through simulation of many benchmark problems that the model developed using PSO based minimization of Wilcoxon norm of errors performs the best compared to that offered by standard squared error norm and other two robust norms [10.2, 10.7, 10.10].

In **Chapter 7** robust adaptive inverse models have been developed using robust cost functions and its BFO based minimization of the cost functions. The inverse model is extensively used in designing equalizers for communication and magnetic media so that ISI is minimized. The BFO is shown to be an efficient learning candidate to design robust inverse models. From the simulation study it is observed that the conventional squared error norm is the least and Wilcoxon norm is the best robust norms to develop inverse models of different types of linear and nonlinear channels.

The identification of Hammerstein plants is a challenging task. This is studied in **Chapter 8** using CPSO and IPSO algorithms. These two new algorithms have been suitably applied for identification task. In this case in the identification task involves response matching of nonlinear static part and matching of parameters of dynamic part of the plants. Identification of standard Hammerstein plants has been carried by simulation study and it is in general observed that the new scheme of identification is superior to GA, AIS and PSO based training schemes [10.8, 10.12, 10.13].

In sensor networks, distributed parameter estimation plays an important role to estimate the temperature, pressure and humidity of a region using local measurements. Distributed linear, nonlinear and robust identifications are therefore have gained importance in sensor network environment. This problem has been studied in **Chapter 9**. We have assumed that the input-output data is linear or nonlinear. Further, the cost function used is squared error or robust norm of errors like Wilcoxon norm. Two new distributed PSO algorithms : incremental PSO and diffusion PSO are developed to identify linear as well as nonlinear plants using local measurements. The new distributed algorithms are applied for parameter estimation of linear plants and response matching of nonlinear plants. It is in general observed that the diffusion PSO algorithm performs better than its counter part. Further the training samples are corrupted with outliers and the identification performance of Wilcoxon norm minimization based model is shown to perform better than that offered by squared error minimization based model. It is observed in case of identification of both linear and nonlinear systems [10.11].

10.2 Further research extension

The work carried out in the present thesis can be extended in many directions. The proposed identification schemes are population based and hence take more training time and are not suitable for online applications. The differential evolution or such fast bio-inspired techniques can be applied to reduce the training time and hence may be applied for online control applications. The bio-inspired training methodology can also be effectively applied to develop efficient forecasting models for prediction of complex financial or other times series like stock market, exchange rates, interest rates, oil price etc. Improved learning algorithms using bio-inspired techniques can be developed to reduce the population size and number of input samples used during training. Further the identification problem can be viewed as a multi-

objective problem by considering the structure complexity and mean square error as two objectives. Then suitable multi-objective bio-inspired techniques may be used to achieve efficient reduced or pruned structure model for identification. Research work can also be carried out on the convergence analysis of bio-inspired optimization algorithms. Little progress is reported in the literature in this direction. Similarly effects of finite register length on the performance of such algorithms is important when such algorithms are implemented in DSP processor or VHDL. Therefore investigation can also be made in this field both using fixed and floating - point arithmetic.

Publications out of the thesis

International Journals

Published

[10.1] Babita Majhi and G. Panda, "Development of Efficient identification Scheme for Nonlinear Dynamic Systems using Swarm Intelligence techniques", **Expert Systems with applications**, Elsevier, vol. 37, issue 1, pp. 556-566, January 2010.

[10.2] **Babita Majhi**, G. Panda and B. Mulgrew, "Robust identification using New Wilcoxon Least Mean Square (WLMS) Algorithm", **IET (IEE) Electronics Letter**, vol. 45, issue 6, pp. 334-335, 12th March 2009.

[10.3] Babita Majhi and G. Panda, "Cascaded Functional Link Artificial Neural Network Model for Nonlinear Dynamic System Identification", **International Journal of Artificial Intelligence and Soft Computing**, vol. 1, Nos. 2/3/4, pp. 223 – 237, 2009.

[10.4] Babita Majhi and G. Panda, "A Hybrid Functional Link Neural Network and Bacterial Foraging Approach for Efficient Identification of Dynamic Systems", **International Journal of Applied Artificial Intelligence in Engineering Systems**, vol. 1, no. 1, pp. 91-104, January-June 2009.

[10.5] **Babita Majhi** and G. Panda, "Identification of IIR Systems using Comprehensive Learning Particle Swarm Optimization", **International Journal of Power and Energy Conversion**, vol. 1, no. 1, pp.105-124, 2009.

[10.6] **Babita Majhi** and G. Panda, "Nonlinear System Identification based on Bacterial Foraging Optimization Technique", **International Journal of Systemics, Cybernetics and Informatics**, ISSN 0973-4864, pp. 44-50, April 2008.

Communicated

[10.7] **Babita Majhi** and G. Panda, "Robust Identification of Nonlinear Complex Systems using Low Complexity ANN and Particle Swarm Optimization Technique", **Expert systems with applications**, Elsevier, May 2009.

[10.8] G. Panda, S. J. Nanda and **Babita Majhi**, "Improved Identification of Hammerstein Plants using new CPSO and IPSO algorithms", **Expert Systems with Applications**, Elsevier, April 2009.

Book Chapter (Accepted)

[10.9] **Babita Majh**i and G. Panda, "Efficient identification of Nonlinear Dynamic Systems using BFO and PSO Techniques", **Book Chapter, Applied Swarm Intelligence,** Springer Series in Studies in Computational Intelligence, June, 2009.

International Conferences

Published

[10.10] **Babita Majhi**, G. Panda and B. Mulgrew, "Robust prediction and identification using Wilcoxon norm and Particle swarm optimization", **17th European Signal Processing Conference (EUSIPCO 2009), Glasgow, UK,** 24-28 August, 2009.

[10.11] **Babita Majhi**, G. Panda and B. Mulgrew, "Nonlinear identification over adaptive networks using distributed PSO algorithms", **IEEE Congress on Evolutionary Computation (CEC 2009), Norway,** pp.2076-2082, May 18-21, 2009.

[10.12] S. J. Nanda, G. Panda and **Babita Majhi**, "Development of Immunized PSO Algorithm and Its Application to Hammerstein Model Identification", **IEEE Congress on Evolutionary Computation (CEC 2009),Norway,** pp.3080-3086, May 18-21, 2009.

[10.13] S. J. Nanda, G. Panda and **Babita Majhi**, "Improved identification of Hammerstein Model based on Artificial Immune System", Proc. of **IEEE International Conference on Emerging Trends in Computing (ICETiC-09), Tamilnadu,** pp. 193-198, 8-10, January 2009.

[10.14] **Babita Majhi** and G. Panda, "Bacterial Foraging based Identification of Nonlinear Dynamics System", Proc. of **IEEE Congress on Evolutionary Computation(CEC-2007), Singapore**, 25-28, September, 2007, pp.1636-1641.

Related Publications

International Journal

Communicated

[10.15] S. J. Nanda, G. Panda and Babita Majhi, "Improved identification of Nonlinear Plants using Artificial Immune System based FLANN model", Communicated to **Engineering Applications of Artificial Intelligence**, Elsevier, **(revised and sent),** October 2009.

International Conferences

Published

[10.16] S. J. Nanda, G. Panda, **Babita Majhi and** P. Tah, "Improved Identification of Nonlinear MIMO Plants using New hybrid FLANN-AIS Model", Proc. of **IEEE International Advance Computing Conference (IACC-09), Patiala,** 6-7, March 2009, pp. 141-146.

[10.17] S. J. Nanda, G. Panda, **Babita Majhi** and P. Tah, "Development of a new optimization algorithm based on Artificial Immune System and its application", Proc. of **IEEE ICIT 2008, Bhubaneswar,** 17-20, December 2008, pp. 45-48.

[10.18] S. J. Nanda, G. Panda and **Babita Majhi, "**Development of Novel Digital Equalizers for noisy nonlinear channel using artificial Immune System", Proc. of **IEEE Region 10 Colloquium and 3rd International Conference on Industrial and Information Systems, IIT Kharagpur,** 8-10, December, 2008, pp. 1-6..

[10.19] S. J. Nanda, G. Panda and **Babita Majhi**, "Improved identification of nonlinear dynamic systems using Artificial Immune System", Proc. of **IEEE INDICON**, **IIT, Kanpur,** 11-13 December, 2008, pp. 268-273.

[10.20] **Babita Majhi,** G. Panda and A. Choubey, "Efficient scheme of identification of Pole-Zero Systems using Particle Swarm Optimization Technique", Proc. of **IEEE Congress on Evolutionary Computation (CEC 2008), Hong Kong,** June 1-6, 2008, pp. 446-451.

[10.21] **Babita Majhi**, G. Panda and H. Thethi, "Development of Efficient Adaptive Nonlinear Channel Equalizers using Fuzzy-Bacterial Foraging Optimization Technique", Proc. of **IEEE Conference on AI Tools in Engineering (ICAITE - 2008), Pune,** 6-8 March, 2008.

[10.22] G. Panda, **Babita Majhi**, D. Mohanty and A. Sahoo, "A GA-based Pruning Strategy and Weight Update Algorithm for Efficient Nonlinear System Identification", Proc. of **International Conference on Pattern Recognition and Machine Intelligence (PReMI' 07), ISI, Kolkata**, 18-22, December 2007, pp. 244-251(Springer-Verlag Berlin Heidelberg 2007).

[10.23] **Babita Majhi** and G. Panda, "Particle Swarm Optimization based Efficient Adaptive Channel Equalizer for Digital Communication", Proc. of **International Conference on Trends in Intelligent Electronics Systems (ties2007), Chennai,** 12-14, November, 2007, pp.23-28.

[10.24] G. Panda, D. Mohanty, **Babita Majhi** and G. Sahoo, "Identification of Nonlinear Systems using Particle Swarm Optimization Technique", Proc. of **IEEE Congress on Evolutionary Computation(CEC-2007), Singapore,** 25-28, September, 2007, pp.3253-3257.

[10.25] **Babita Majhi** and G. Panda, "Recovery of Digital Information using Bacterial Foraging Optimization based Nonlinear Channel Equalizers", Proc. of **IEEE 1st International Conference on Digital Information Management (ICDIM-2006),Bangalore,** India, 6-8th December, 2006, pp. 367-372.

[10.26] G. Panda, **Babita Majhi** and S. Mishra, "Nonlinear System Identification using Bacterial Foraging based Learning", Proc. of **IET 3rd International Conference on Artificial Intelligence in Engineering Technology (ICAIET-2006), Malaysia,** 22-24, November, 2006, pp. 120-125.

[10.27] **Babita Majhi**, G. Panda and A. Choubey, "On The Development of a new Adaptive Channel Equalizer using Bacterial Foraging Optimization Technique", Proc. of **IEEE Annual India Conference (INDICON-2006), New Delhi**, India, 15th-17th September, 2006, pp. 1-6.

[10.28] G. Panda, **Babita Majhi**, D. Mohanty and A. Choubey, "Development of GA Based Adaptive Techniques for Nonlinear System Identification", Proc. of **International Conference on Information Technology (ICIT-2005), Bhubaneswar, I**ndia, 20-23, December, 2005, pp. 198-204.

National Conferences

Published

[10.29] T. Panigrahi, G. Panda and Babita Majhi, "Collaborative Signal Processing in Distributed Wireless sensor Network", Proc. of **National Conference on Advances in Computational Intelligence Applications (NCACI 2009), Bhubaneswar**, 20-22 March, 2008.

[10.30] U. K. Sahoo, G. Panda and **Babita Majhi**, "Distributed regression : an efficient framework for modeling sensor network data", Proc. of **National Conference on Advances in Computational Intelligence Applications (NCACI 2009), Bhubaneswar,** 20-22 March, 2008.

[10.31] P. M. Pradhan, B. Mulgrew, G. panda and **Babita Majhi**, "Layout optimization of Wireless sensor network using NSGA-II", **National Conference on Advances in Computational Intelligence Applications (NCACI 2009), Bhubaneswar,** 20-22 March, 2008.

[10.32] D. Mohanty, **Babita Majhi** and G. Panda, "Recovery of digital data using real coded genetic algorithm", Proc. of **National Seminar on Soft Computing Techniques & applications (SCTA-2007), Bhubaneswar**, India, 7th April 2007, pp. 39-44.

[10.33] G. Panda, D. Mohanty and **Babita Majhi**, *"*Development of novel digital nonlinear channel equalizers using Particle Swarm Optimization technique*"*, **National Conference on Data Mining and e-Goverence, Bilaspur**, India, 17th Feb., 2007, pp. 25 (abstract).

[10.34] G. Panda, **Babita Majhi** and A. Choubey, *"*Nonlinear Adaptive Channel Equalization using GA based Technique*"*, Proc. of **National Conference on Soft Computing and Machine Learning for Signal Processing, Control, Power and Telecommunications (NCSC-2006), Bhubaneswar,** India, 24-26, March, 2006, pp. 32(abstract).

[10.35] G. Panda, D. Mohanty and **Babita Majhi,** "An Adaptive Genetic Algorithm for System Identification : a faster approach", Proc. of **National Conference on Soft Computing Techniques for Engineering Applications (SCT-2006), NIT Rourkela,** India, 24-26, March, 2006, pp. 329-336.

[10.36] G. Panda, **Babita Majh**i, D. Mohanty, A. Choubey and S. Mishra, "Development of Novel Digital Channel Equalisers using Genetic Algorithms", Proc. of **National Conference on Communication (NCC-2006), IIT Delhi,** India, 27-29,January, 2006, pp.117-121.

[10.37] G. Panda, **Babita Majhi**, D. Mohanty and B. N. Biswal, "Application of Adaptive Genetic Algorithm in Nonlinear System Identification", Souvenir of **National Conference of Cyber Security, Data Mining and ICT for Society, Bilaspur**, India, 18-19, January, 2006, pp. 6 (abstract).